ON GENERALIZED LDPC CODES FOR ULTRA RELIABLE COMMUNICATIONS

by

YANFANG LIU

in partial fulfillment of the requirements for the degree of Doctor in Multimedia and Communication

UNIVERSIDAD CARLOS III de MADRID

Advisors:

PABLO OLMOS MARTÍNEZ

TOBIAS KOCH

January 31, 2019

ACKNOWLEDGEMENT

Firstly, I would like to express my sincere gratitude to my advisor Dr. Pablo Olmos Martínez and Dr. Tobias Koch for the continuous support of my PhD study and related research, for their patience, motivation, and immense knowledge. His guidance helped me in all the time of research and writing of this thesis.

Besides my advisor, I would like to thank my thesis committee: Prof. JuanJo Murillo, Prof. Matilde Sánchez Fernández and Prof. Javier Valls, for their insightful comments and encouragement, but also for the hard question which invented me to widen my research from various perspectives.

My sincere thanks also goes to Dr. David G. M. Mitchell, who provided me an opportunity to join his team as intern, and keep collaborating after my internship. Without his precious support it would not be possible to conduct this research.

Moreover, I would like to express my special thanks to my friends and colleagues in my lab. As a foreigner who does not speak Spanish, life is never easy. However, my lab mates helped me a lot to figure out all problems during my stay in Madrid. In particular, Grace Villacrés helped me made and answered so many phone calls in Spanish. Cheers for all the fun we have had in the last four years.

Last but not the least, I would like to thank my family: my mom, my younger brother, my husband and my cat, for supporting me spiritually throughout writing this thesis and my life in general.

PUBLISHED AND SUBMITTED CONTENT

The technical contributions presented in this thesis have been published in the following journal and international conference papers:

1 Yanfang Liu, Pablo M. Olmos, and Tobias Koch. A probabilistic Peeling Decoder to efficiently analyze generalized LDPC codes over the BEC. Submitted to IEEE Transactions on Information Theory (2nd review, available: `https://arxiv.org/pdf/1709.00873.pdf`). It is wholly included in the Chapter 2 of the Thesis, and partly included in the Chapter 4 of the Thesis. The material from this source included in this thesis is not singled out with typographic means and references.

2 Yanfang Liu, Pablo M. Olmos, and Tobias Koch. On LDPC code ensembles with generalized constraints. Proceedings of 2017 IEEE Information Theory (ISIT), 2017 IEEE pp. 371-375, Acchen, Germany, June, 2017. It is partly included in the Chapter 2 of the Thesis. The material from this source included in this thesis is not singled out with typographic means and references.

3 Yanfang Liu, Pablo M. Olmos, and David G. M. Mitchell. On Generalized LDPC Codes for 5G Ultra Reliable Communication. Proceedings of the 2018 IEEE Information Theory Workshop (ITW), Guangzhou, China, November, 2018. It is partly included in the Chapter 3 of the Thesis. The material from this source included in this thesis is not singled out with typographic means and references.

4 Yanfang Liu, Pablo M. Olmos, and David G. M. Mitchell. Generalized LDPC Codes for Ultra Reliable Low Latency Communication in 5G and Beyond. Accepted for publication in IEEE Access, Special Issue on Advances in Channel Coding in 5G and Beyond. November 2018. It is wholly included in the Chapter 3 of the Thesis, and partly included in the Chapter 4 of the Thesis.

The material from this source included in this thesis is not singled out with typographic means and references.

Abstract

Ultra reliable low latency communication (URLLC) is an important feature in future mobile communication systems, as they will require high data rates, large system capacity and massive device connectivity [11]. To meet such stringent requirements, many error-correction codes (ECC)s are being investigated; turbo codes, low density parity check (LDPC) codes, polar codes and convolutional codes [70, 92, 38], among many others. In this work, we present generalized low density parity check (GLDPC) codes as a promising candidate for URLLC.

Our proposal is based on a novel class of GLDPC code ensembles, for which new analysis tools are proposed. We analyze the trade-off between coding rate and asymptotic performance of a class of GLDPC codes constructed by including a certain fraction of generalized constraint (GC) nodes in the graph. To incorporate both bounded distance (BD) and maximum likelihood (ML) decoding at GC nodes into our analysis without resorting to multi-edge type of degree distribution (DD)s, we propose the probabilistic peeling decoding (P-PD) algorithm, which models the decoding step at every GC node as an instance of a Bernoulli random variable with a successful decoding probability that depends on both the GC block code as well as its decoding algorithm. The P-PD asymptotic performance over the BEC can be efficiently predicted using standard techniques for LDPC codes such as Density evolution (DE) or the differential equation method. We demonstrate that the simulated P-PD performance accurately predicts the actual performance of the GLPDC code under ML decoding at GC nodes. We illustrate our analysis for GLDPC code ensembles with regular and irregular DDs.

This design methodology is applied to construct practical codes for URLLC. To this end, we incorporate to our analysis the use of quasi-cyclic (QC) structures, to mitigate the code error floor and facilitate the code very large scale integration (VLSI) implementation. Furthermore, for the additive white Gaussian noise (AWGN) channel, we analyze the complexity and performance of the message passing decoder with various update rules (including standard full-precision sum-

product and min-sum algorithms) and quantization schemes. The block error rate (BLER) performance of the proposed GLDPC codes, combined with a complementary outer code, is shown to outperform a variety of state-of-the-art codes, for URLLC, including LDPC codes, polar codes, turbo codes and convolutional codes, at similar complexity rates.

Extended Abstract

URLLC is one of the key factors in the fifth-generation (5G) of cellular mobile communications. To meet expectations for the demanding digital industry and latency-sensitive services, it requires high data rates, large system capacity and massive device connectivity [11]. Many error-correction codes (ECCs) are being investigated to meet the stringent requirements of URLLC: turbo codes, LDPC codes, polar codes, and convolutional codes [70, 92, 38], among many others. Beyond any doubt, LDPC codes, Polar codes and their variants will be included in other new standards in the future. The state-of-the-art achievements of LDPC codes show capacity achieving performance.

However, this requires large block length. For short block length, the error floor problem which refers to the problem that the bit error rate (BER) performance curve does not decrease as the SNR increases [33], becomes relevant. Under iterative message passing decoding, the error floor of LDPC codes depends on a number of structural properties of the codes and tanner graphs, such as girth, minimum weight, weight distribution of pseudocodewords [94, 84, 99], and is higher than the one under Maximum a posterior probability (MAP) decoding. Thus, the design of error correcting codes with short block length and good performance under practical iterative decoding, as required for next-generation wireless communication systems, is still very challenging. Generalized low-density parity-check (GLDPC) codes are a promising class of codes for low latency communication. To design codes for URLLC, we propose a novel class of GLDPC code ensembles, for which new analysis tools are proposed.

We analyze the trade-off between coding rate and asymptotic performance of a class of GLDPC codes constructed by including a certain fraction of generalized constraint (GC) nodes in the graph. The rate of the GLDPC ensemble is bounded using classical results on linear block codes, namely Hamming bound and Varshamov bound. We study the impact of the decoding method used at GC nodes. To incorporate both bounded-distance (BD) and Maximum Likelihood

(ML) decoding at GC nodes into our analysis without resorting to multi-edge type of degree distributions (DDs), we propose the probabilistic peeling decoding (P-PD) algorithm, which models the decoding step at every GC node as an instance of a Bernoulli random variable with a successful decoding probability that depends on both the GC block code as well as its decoding algorithm. The P-PD asymptotic performance over the BEC can be efficiently predicted using standard techniques for LDPC codes such as density evolution (DE) or the differential equation method. Furthermore, for a class of GLDPC ensembles, we demonstrate that the simulated P-PD performance accurately predicts the actual performance of the GLPDC code under ML decoding at GC nodes. We illustrate our analysis for GLDPC code ensembles with regular and irregular DDs.

This design methodology is applied to construct practical codes for URLLC. To this end, we incorporate to our analysis the use of quasi-cyclic structures, to mitigate the code error floor and facilitate the code VLSI implementation. Furthermore for the AWGN channle, we analyze the complexity and performance of the message passing decoder with various update rules (including standard full-precision sum-product and min-sum algorithms) and quantization schemes. The block error rate (BLER) performance of the proposed GLDPC codes, combined with a complementary outer code, is shown to outperform a variety of state-of-the-art codes for URLLC, including LDPC codes, polar codes, turbo codes and convolutional codes, at similar complexity rates.

Contents

List of Acronyms **7**

1 Introduction **9**

1.1 Motivation 9

1.2 Low Density Parity-Check Codes 13

1.2.1 Background 13

1.2.2 LDPC ensemble Notation 13

1.2.3 Decoding over the Binary Erasure Channel 15

1.2.4 LDPC decoding for the additional white Gaussian noise channel 20

1.2.5 LDPC finite length performance 23

1.3 Generalized LDPC Codes 26

1.3.1 Background 26

1.3.2 GLDPC decoding algorithms over the BEC 28

1.3.3 GLDPC decoding algorithms over general channels 32

1.3.4 Contributions of the proposed asymptotic analysis technique in this work 33

2 Probabilistic Peeling Decoder Analysis of GLPDC codes **35**

2.1 GLDPC ensembles with increasing fraction of GC nodes 38

2.1.1 Degree distribution 38

2.1.2 The coding rate of the $\mathcal{C}_{J,K,\nu}$ ensemble 39

2.1.3 Growth rate of the weight distribution of the $\mathcal{C}_{J,K,\nu}$ ensemble 42
2.2 Probabilistic Peeling Decoding over the BEC 43
2.2.1 Comparing the P-PD and ML-PD performances by Monte Carlo simulation 46
2.3 Asymptotic analysis 47
2.3.1 An upper bound on the iterative-decoding threshold 51
2.4 Analysis of the $\mathcal{C}_{J,K,\nu}$ ensemble under P-PD 52
2.4.1 Results for $(2, 6)$ and $(2, 7)$ base DDs 53
2.4.2 Results for higher-density base DDs 57
2.5 Selecting specific component codes 58
2.5.1 Growth Rate of the Weight Distribution 60
2.5.2 Extension to irregular GDLPC code ensembles 62
2.6 Random puncturing 64
2.7 Doubly-generalized LDPC codes 65
2.7.1 Decoding via P-PD 67
2.7.2 Degree Distribution and Asymptotic Analysis 68
2.7.3 Results for the $(3, 6)$ and $(3, 7)$ base DDs 69
2.8 Applications of GLDPC codes 71

3 Generalized LDPC Codes for Ultra Reliable Low Latency Communication 73
3.1 Parameters of the designed GLDPC codes 76
3.2 Practical GLDPC Code Design for 5G URLLC 80
3.2.1 Code Design Parameters 80
3.2.2 QC Graph Lifting 81
3.2.3 Location of the GC Nodes 84
3.2.4 Target Coding Rates 86
3.3 GLDPC Message Passing 89
3.3.1 Min-sum Decoding Algorithms 90
3.3.2 Finite-precision with Uniform Quantization 92
3.3.3 The Effects of Different Maximum Iteration Numbers . . . 92

3.3.4 Decoding Complexity 93
3.4 Experimental Results 96

4 Conclusions and Future work **101**
4.1 Conclusions 101
4.2 Future lines of work 103

A Wormald's Theorem and the proof of Theorem 4 **105**
A.1 Wormald's Theorem and the proof of Theorem 4 105
A.1.1 Wormald's theorem [103] 105
A.1.2 Expected graph evolution under P-PD 107

B Proof of Theorem 5 **115**
B.1 Proof of Theorem 5 115

C Generator matrices of reference Codes **119**
C.1 Generator matrices of reference Codes 119

D Update Rules of the GC Nodes for the (6,3) Shortened Hamming code **123**
D.1 Update Rules of the GC Nodes for the (6,3) Shortened Hamming code 123

References **125**

List of Acronyms

5G	fifth-generation
AWGN	additive white Gaussian noise
BCH	Bose Chaudhuri Hocquenghem
BD	bounded distance
BD-PD	PD with BD decoding at GC nodes
BER	bit error rate
BLER	block error rate
BP	Belief Propagation
CA-SCL	cyclic redundancy check (CRC)-aided SCL
CMMB	China mobile multimedia broadcasting
CRC	cyclic redundancy check
DD	degree distribution
DE	density evolution
eBCH	extended BCH
ECC	error-correction codes
eMBB	Enhanced Mobile Broadband
EXIT	extrinsic information transfer

GC	generalized constraint
GLDPC	generalized low density parity check
GV	generalized variable
LDPC	low density parity check
LLR	log-likelihood ration
LR	likelihood ration
M2M	machine-to-machine
MAP	maximum a posterior probability
MET	multi-edge-type
ML	maximum likelihood
ML-PD	PD under ML-decoded component codes
mMTC	Massive Machine-Type Communication
MS	min-sum
MSA	min-sum algorithm
OSD	ordered statistics decoder
PD	peeling decoding
P-PD	probabilistic peeling decoding
QC	quasi-cyclic
QPSK	quadrature phase shift keying
RV	regular variable
SC	Successive Cancellation
SC-LDPC	Spatially-coupled LDPC

SCL	Successive-Cancellation List
SISO	soft-input soft-output
SPA	sum-product algorithm
SPC	single parity check
STC	stability condition
URLLC	ultra reliable low latency communication
VLSI	very large scale integration
VN	variable node

1 Introduction

1.1 Motivation

Ultra Reliable Low Latency Communication (URLLC) is one of the key factors in future cellular mobile communications, including the fifth generation (5G). Three main service categories in 5G have been defined by the third generation partnership project [90]. The first is Enhanced Mobile Broadband (eMBB), which is the service category, designed for services that have high requirements for bandwidth, such as virtual reality and augmented reality. The second is Massive Machine-Type Communication (mMTC) that supports a massive number of devices characterized by ultra-low power consumption to increase the device lifetime. The third category is Ultra Reliable Low Latency Communication (URLLC), which focuses on delay sensitive applications and services, such as assisted and automated driving, remote management, fault detection, frequency and voltage control in smart grids [11]. As

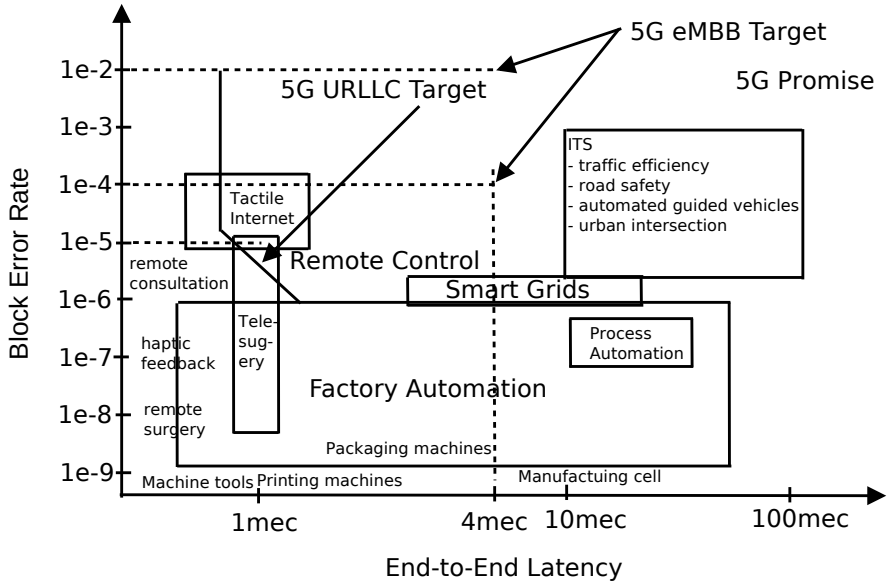


Figure 1.1: Reliability and latency requirements for different URLLC services (Figure borrowed from [90])

shown in Figure 1.1, URRLC requires very low error rates: factory automation and tele-surgery have reliability requirements of 10^{-9} with an end-to-end latency of less than 1ms.

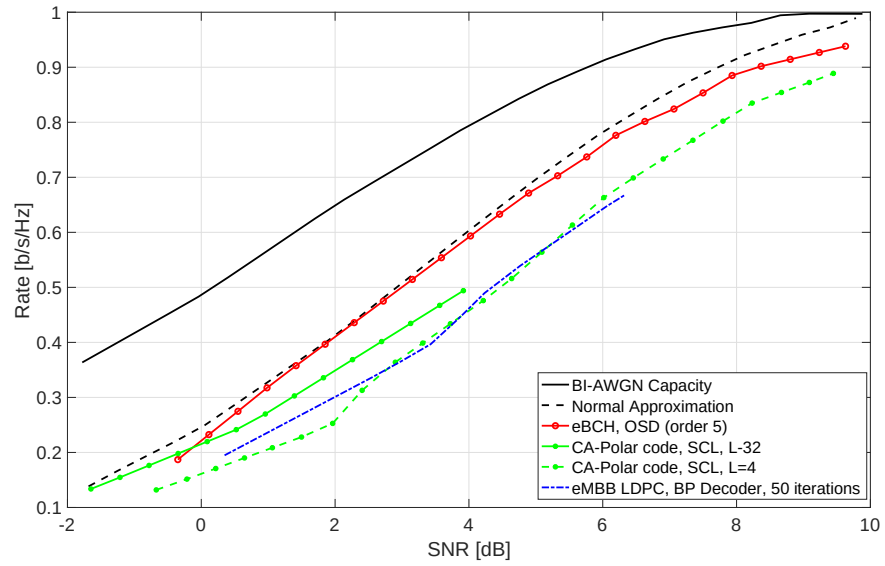
Other services, such as smart grids or the tactile internet have reliability requirements of 10^{-6} at latencies between 1ms and 100ms [90]. In this thesis, we focus on designing error-correction codes (ECCs) to meet the requirements of URLLC. Many ECCs are being investigated to this end. For instance, turbo codes, LDPC codes, polar codes, or convolutional codes [70, 92, 38]. In addition, LDPC codes have been selected for the EMBB data channels for 5G New Radio, while Polar codes have been chosen for the corresponding control channel [90].

The state-of-the-art achievements of LDPC codes show capacity-achieving performance when the block length is large. In the finite length regime, LDPC codes suffer from non-negligible error floors [51]. The error floor of LDPC codes depends on a number of structural properties of the Tanner graph they are designed on, such as girth and distance codeword spectrum [94, 84, 99]. Furthermore, under iterative message passing decoding, the error floor problem gets worse because of

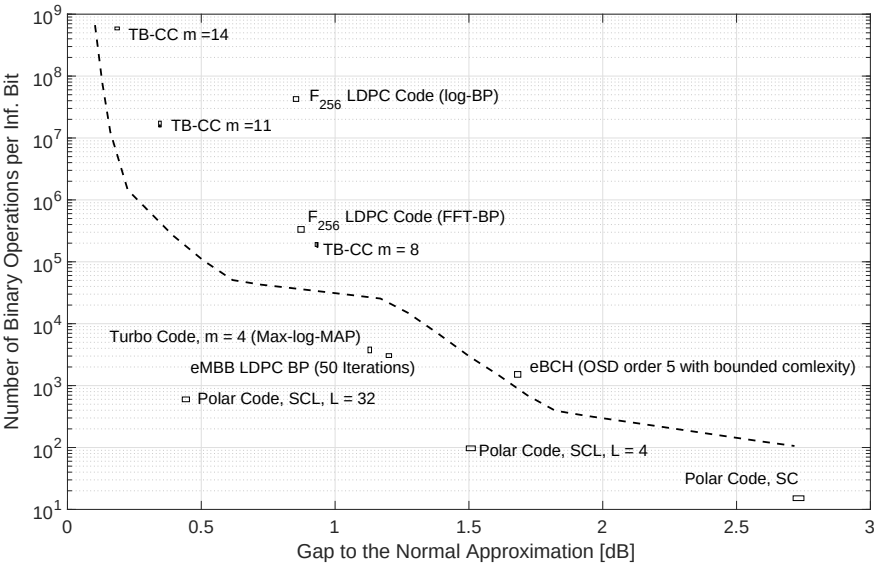
"stopping sets" [87, 41, 76, 84], related to pseudocodewords [16], which is particularly noticeable in the short block length regime [98, 19]. Thus, error-correcting codes of short length that have a good performance under iterative message passing decoding, as required for next-generation wireless communication systems, is a great challenge.

Another class of ECCs that recently received a lot of attention are Polar codes [35], which is a family of capacity-achieving error-correction linear block codes, constructed on a recursive concatenation of a short kernel code, which transforms the physical channel into virtual outer channels [58]. In [7], a comparison between Polar codes, LDPC code and Turbo decoders for existing communications standards is investigated, both in terms of error-correction performance and hardware efficiency. Belief Propagation (BP) decoding algorithm, Successive Cancellation (SC) decoding algorithm and Successive-Cancellation List (SCL) decoding algorithms are considered. Figure 1.2 (a) compares the performance of such coding schemes. It turns out that both the BP and SC decoding are not powerful enough to approach the fundamental limits in the finite block length regime at moderate SNR values. As we can see, extended BCH (eBCH) with ordered statistics decoder (OSD) codes outperform all other existing codes at all SNRs. However, OSD is a quasi-ML decoding method, with large computational complexity. Figure 1.2 (b) shows the trade-off between performance and complexity. Among the two sets of Polar codes with cyclic redundancy check (CRC)-aided SCL (CA-SCL) with list sizes 4 and 32, the decoder with list size 32 is significantly better, but has a significantly larger decoding complexity. The performance of short block length LDPC codes designed for enhanced mobile broadband (eMBB) under BP decoding is slightly better than the CA-Polar code with SCL decoding of list size 4.

We conclude that novel approaches are needed to meet the demands of URLLC systems, we put forward generalized LDPC (GLDPC) codes as strong candidates for URLLC, able to outperform both LDPC codes and Polar codes at similar complexity. In the rest of the chapter, we briefly review the basic concepts of LDPC codes, GLDPC codes, and message-passing decoding.



(a)



(b)

Figure 1.2: In figure (a), we show the rates versus SNR for different error correcting codes with block length 128 at BLER = 10^{-4}. As benchmark, we further show the capacity of the binary-input AWGN channel and the corresponding normal approximation at BLER = 10^{-4}. In figure (b) we plot the algorithmic complexity versus performance for different rate 1/2 channel codes with block length 128 at BLER = 10^{-4}. (Figure borrowed from [90])

1.2 Low Density Parity-Check Codes

1.2.1 Background

Low density parity-check (LDPC) codes are a class of linear block codes, which were originally proposed by Gallager in 1960's [26], but were not quite developed for a long time, due to the fact that the existing hardware could not support effective decoder implementation [67]. 35 years later, MacKay, Luby, Urbanke and Richardson, among others [55], [53], [3] rediscovered LDPC codes and proved that LDPC codes are capable of closely approaching the channel capacity under feasible low-complexity iterative decoding. Since then, LDPC codes have been included in many communication standards, such as IEEE 802.6, IEEE 802.20, IEEE 802.3, DBV-RS2 and China mobile multimedia broadcasting (CMMB).

Compared to turbo codes [21, 91], LDPC codes have performance and complexity advantages, particularly at high code rates [5, 23]. Among others, we can mention that the number of iterations for turbo codes is fixed in the decoder, while the LDPC decoder easily incorporates an early stopping rule, which significantly reduces the number of iterations. Moreover, the LDPC decoder can be implemented in a parallel scheme, which is important when considering large block lengths and low latency [23]. Furthermore, several works have reported that LDPC codes have favorable error floors compare to turbo codes [84]. In the following we give the notations and parameters used to define and analyze the LDPC codes.

1.2.2 LDPC ensemble Notation

LDPC codes are linear block codes specified by parity-check matrices containing mostly 0's and only a small number of 1's [27]. For instance, a (N, J, K) regular

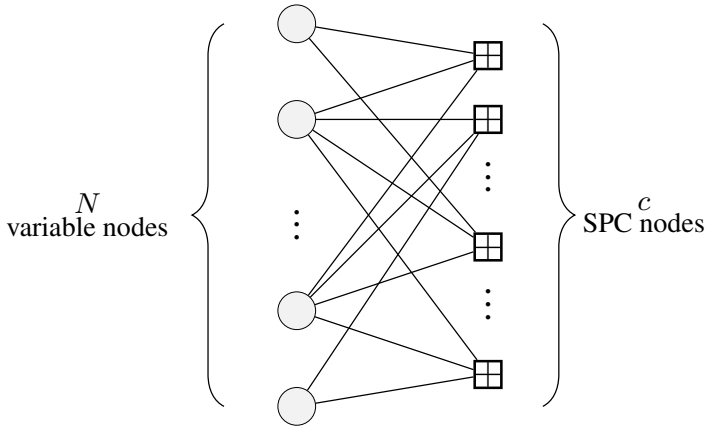


Figure 1.3: Tanner graph of a LDPC code. N is the number of variable nodes and c is the total number of check nodes in the Tanner graph.

LDPC code has block length N with a parity-check matrix as follows

$$\mathbf{H} = \begin{bmatrix} 1 & 1 & 1 & 0 & 1 & \dots & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & \dots & 0 & 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 & \dots & 0 & 0 & 1 & 0 & 0 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \dots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & 0 & 0 & 0 & 1 & \dots & 0 & 0 & 1 & 0 & 0 & 1 \end{bmatrix},$$

where each column contains a small fixed numbers, J, of 1's, and each row contains a small fixed number, K, of 1's [27]. The code can be represented by a Tanner graph, shown in Figure 1.3, where the left part in the graph represents the N variable node (VN)s and the right part represents the c single parity check (SPC) nodes. The lines between VN and SPC are called edges. The degree of a VN is the number of edges adjacent to it. Similarly, the degree of SPC is the number of edges adjacent to this node. If all VNs have constant degree J and all SPC nodes have constant degree K, then the LDPC code is said to be regular. Otherwise, it is an irregular LDPC code. We denote by $\mathtt{E}$ the number of edges in the Tanner graph. Given these definitions, the degree distribution (DD) of the LDPC code is characterized as follows. The vector $\overline{\lambda} = (\lambda_1, \lambda_2, ..., \lambda_J)$ is the *left* DD, where λ_i represents the fraction of edges (w.r.t. $\mathtt{E}$) connected to a variable node of degree

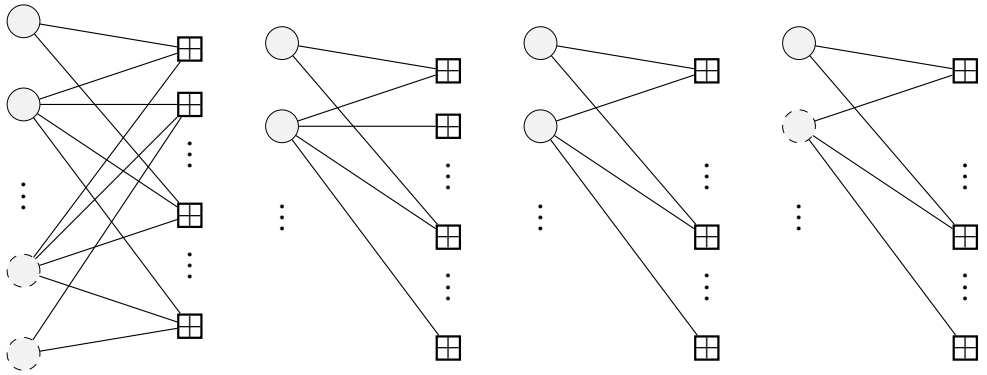


Figure 1.4: The peeling decoding process.

i. Therefore, $\overline{\lambda}$, N and $\mathtt{E}$ are related as follows [86]

$$N = \mathtt{E} \sum_{i=1}^{J} \lambda_i / i. \tag{1.1}$$

The *right* DD is defined by vector $\overline{\rho}_p = (\rho_{p1}, \rho_{p2}, ..., \rho_{pK})$, where ρ_{pj} denotes the fraction of edges (w.r.t. $\mathtt{E}$) connected to a SPC node that has degree j. Note that the LDPC graph is specified in terms of fractions of *edges*, not *nodes*, of each degree. The average left degree of the graph is $\sum_i i\lambda_i$, the average right degree is $\sum_j j\rho_{pj}$, and so the design coding rate is [86]

$$R = 1 - \frac{\sum_i i\lambda_i}{\sum_j j\rho_{pj}}. \tag{1.2}$$

1.2.3 Decoding over the Binary Erasure Channel

A binary erasure channel (BEC) is a communication channel model extensively used in coding theory and information theory. It was first introduced by Elias in 1954 as a toy example [86]. In this channel, the receiver either receives a transmitted binary bit (0 or 1) correctly or an erasure (?), with the probability ϵ, as shown in Figure 1.5. The BEC is often used because it is one of the simplest noisy channels to analyze. It is also a good communication channel model for packed communications.

Iterative decoding of LDPC codes over the BEC can be performed by peeling decoding (PD) algorithms [50, 57, 75], which iteratively remove variable nodes

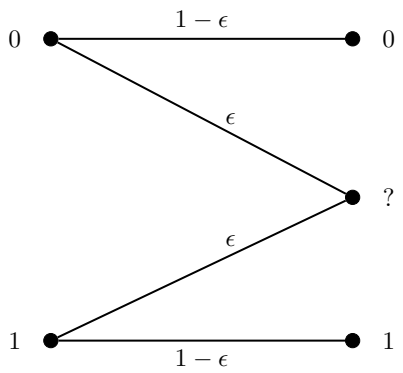


Figure 1.5: Binary erasure channel with channel erasure ϵ.

Algorithm 1 PD

Remove from the Tanner graph of the LDPC code all variable nodes with indexes in $\Gamma_{\boldsymbol{y}}$.

Construct Ψ, the index set of check nodes that correspond to degree-one SPC nodes.

repeat

1) Select at random a member of Ψ.

2) Remove from the Tanner graph the check node with the index drawn in Step 1). Further, remove the connected variable node, and all attached edges.

3) Update Ψ.

until All variable nodes have been removed (successful decoding) or $\Psi = \emptyset$ (decoding failure).

whose value is known from the Tanner graph. We illustrate the PD process in Figure 1.4. Suppose we use a random LDPC code of block length N to transmit over a BEC(ϵ),for which each of the N code bits is erased with probability ϵ. Without loss of generality, we assume that the all-zero codeword is transmitted, hence the received vector $\boldsymbol{y}$ belongs to the set $\{0, ?\}^N$, where ? denotes an erasure. Let $\Gamma_{\boldsymbol{y}} \subseteq \{1, \ldots, N\}$ be the index set of the bits correctly received, namely $y_i = 0$ for all $i \in \Gamma_{\boldsymbol{y}}$. After the BEC transmission, first of all, the PD removes all corrected received VNs along with all adjacent edges from the tanner graph, as shown in

Figure 1.4 (a). In the next step, the PD picks at random a degree-1 SPC node. Note that a degree-1 SPC node represents a parity-check equation in which we know all variables in the parity equation but one. The PD removes the degree-1 SPC node from the graph, along with the adjacent edge, as shown in Figure 1.4 (b) and Figure 1.4 (c), respectively. Furthermore, the VN connected with this edge is decodable and will be removed from the graph, along with all adjacent edges, as shown in Figure 1.4 (d). Continuously, the PD repeats this procedure until there is no VN left in the graph, which corresponds to a decoding *success*. If there is no degree-1 SPC left in the graph before successful decoding, then we say that there was a decoding *failure*. Thus, the key point of a successful decoding is to keep having degree-1 SPC nodes on the residual graphs that are sequentially generated during the PD process. We summarize the PD in Algorithm 1.

As a result, the PD decoding process yields a sequence of graphs. As shown in [50], the mean of such sequence of residual graphs coincides with the asymptotic (in the blocklength) average evolution of the ensemble. This asymptotic evolution can be computed by solving a particular set of differential equations [50]. The threshold of the LDPC code ensemble is given by the maximum BEC parameter for which there is always at least one degree-1 SPC node until decoding success.

PD asymptotic analysis

In the following, we define the notation used to predict asymptotic DD evolution of the residual LDPC tanner graph and introduce the differential equation technique proposed in [50]. Suppose we use a LDPC code ensemble with maximum left degree J and right degree K for transmission over the BEC with parameter ϵ. The total number of edges in the LDPC graph is $\mathtt{E}$. Given the residual graph at the ℓ-th iteration of the PD algorithm, let $L_i^{(\ell)}$ represent the number of edges that have left degree i at iteration ℓ, and let $R_{pj}^{(\ell)}$ represent the number of edges that have right degree j at iteration ℓ. Using Wormald's theorem (See Appendix A.1.1), in [50] the authors prove that the DD of the residual graph at iteration ℓ

under PD algorithm converges with $\mathtt{E}$ to

$$L_i^{(\ell)}/\mathtt{E} \to l_i^{(\tau)}, \quad i \in \{1, \ldots, J\} \tag{1.3}$$

$$R_{pj}^{(\ell)}/\mathtt{E} \to r_{pj}^{(\tau)}, \quad j \in \{1, \ldots, K\} \tag{1.4}$$

where the notion of convergence is given in Appendix A.1.1 and $\tau = \frac{\ell}{\mathtt{E}} \in [0, \sum_{i=1}^{J} l_i^{(\tau)}/i]$. In (1.3) and (1.4), ($l_i^{(\tau)}$, $r_{pj}^{(\tau)}$) are the solutions to the following system of differential equations:

$$\frac{\mathrm{d}l_i^{(\tau)}}{\mathrm{d}\tau} = -\frac{i l_i^{(\tau)}}{e^{(\tau)}}, \tag{1.5}$$

$$\frac{\mathrm{d}r_{pj}^{(\tau)}}{\mathrm{d}\tau} = (r_{p(j+1)}^{(\tau)} - r_{pj}^{(\tau)})\frac{j(a^{(\tau)} - 1)}{e^{(\tau)}} - \mathbb{I}[j = 1], \tag{1.6}$$

where $\mathbb{I}[\cdot]$ denotes the indicator function, and

$$e^{(\tau)} = \sum_{i=1}^{J} l_i^{(\tau)} = \sum_{j=1}^{K} r_{pj}^{(\tau)}, \tag{1.7}$$

$$a^{(\tau)} = \sum_{i} i l_i^{(\tau)}/e^{(\tau)}, \tag{1.8}$$

are, the fraction of remaining edges in the graph ate time τ and the average left degree, respectively. The initial conditions of the system of differential equations (1.5)-(1.6) are given by

$$l_i^{(0)} = \epsilon\lambda_i, \tag{1.9}$$

$$r_{pj}^{(0)} = \sum_{\alpha \geq j} \rho_{p\alpha} \binom{\alpha - 1}{j - 1} \epsilon^j (1 - \epsilon)^{\alpha - j} \tag{1.10}$$

for $i = 1, \ldots J$, $j = 1, \ldots, K$ [50]. The asymptotic PD threshold is computed by the largest channel parameter ϵ for which $r_{p1}^{(\tau)} > 0$ during the whole decoding process, $\tau \in [0, \sum_{i=1}^{J} \frac{l_i}{i}]$. For instance, in Figure 1.6 we plot the fraction $r_{p1}(\tau)$ of edges with right degree one as a function of the $e(\tau)$, the fraction of edges remaining in the residual LDPC graphs, when the DD of the LDPC codes correspond to (3,6)-regular, (4,8)-regular and (5,10)-regular LDPC code ensembles. The quantities $r_{p1}(\tau)$ and $e(\tau)$ are computed by numerical integration from (1.5) and (1.6) using Luby's method. The thresholds computed by this differential technique are

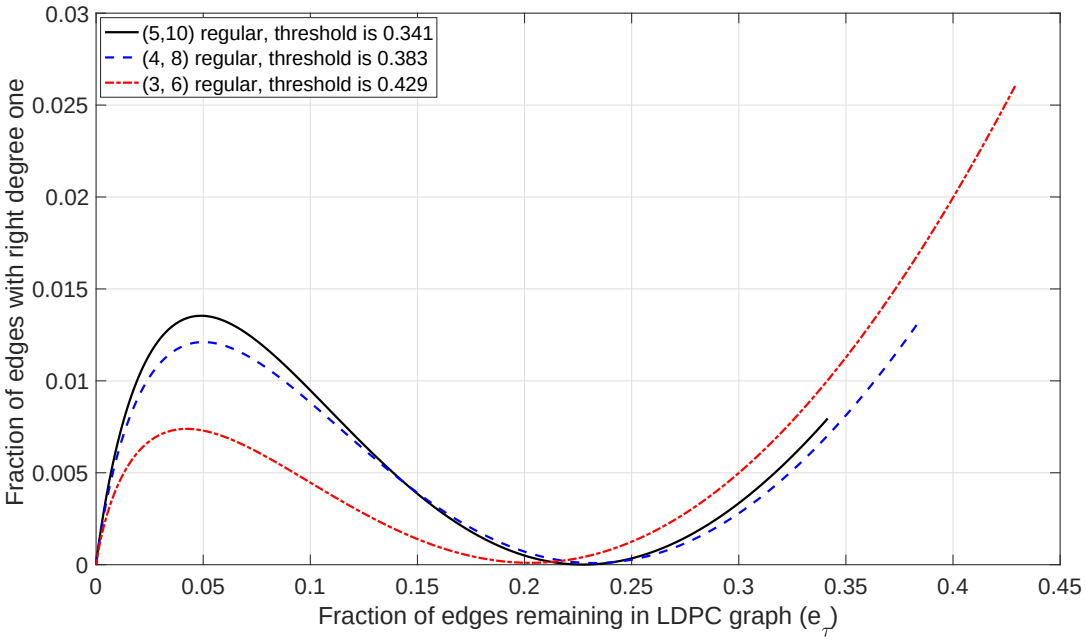


Figure 1.6: Fraction of edges with right degree one as a function of the fraction of edges remaining in the graph.

equivalent to those computed by other methods, such as density evolution (DE) and EXIT charts [85].

Two classes of LDPC codes have been shown to present capacity-achieving thresholds. On the one hand, irregular LDPC codes, which were first proposed in [50, 83] contain both low-degree VNs (mostly degree-2 VNs) and VNs of very high degrees. However, irregular LDPC codes have severe error floors [83]. On the other hand, capacity-achieving LDPC code ensembles can also be obtained by spatially-coupling LDPC block codes with regular DDs [32]. In [62], the authors constructed protograph-based Spatially-coupled LDPC (SC-LDPC) codes by coupling together a series of disjoint, or uncoupled, LDPC code Tanner graphs into a single coupled chain. This opened up a new way to construct capacity-achieving codes for memoryless binary-input symmetric-output channels with low-complexity BP decoding. A scaling law to predict the error probability of finite-length spatially coupled LDPC codes over BEC is proposed in [74]. It is shown that, while SC-LDPC codes do not suffer from error floors, their performance is severely degraded in the

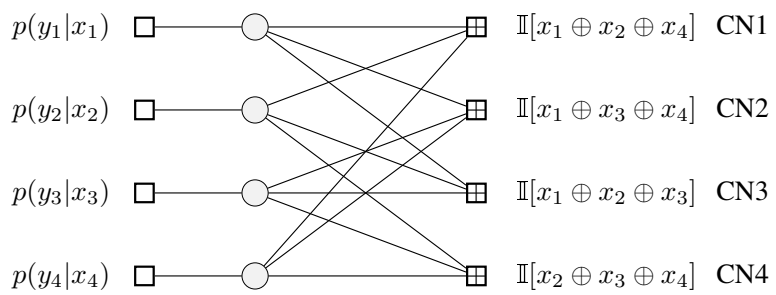


Figure 1.7: Factor graph of a linear block code.

finite block length regime, hence block lengths in the order of at least thousands of bits are required to improve upon uncoupled LDPC codes.

1.2.4 LDPC decoding for the additional white Gaussian noise channel

For the AWGN channel, LDPC decoding is performed by means of sub-optimal sum-product algorithm (SPA), also known as iterative belief propagation (BP) [86, 88]. In [40], the authors use factor graphs to illustrate the operation of the SPA in a straightforward way. In Figure 1.7 we show the factor graph of a linear block code. The left factor nodes are channel likelihoods, $p(y_i|x_i)$, and the right nodes are parity check functions.

SPA is a message passing algorithm, in which variable nodes and factor nodes exchange probability messages at every iteration. Initially, every variable computes its probability using the channel likelihood. Then, the probabilities p_{i0} and p_{i1} corresponding to the ith variable node are given by

$$p_{i0} = \frac{p(y_i|0)}{p(y_i|0) + p(y_i|1)} \qquad p_{i1} = \frac{p(y_i|1)}{p(y_i|0) + p(y_i|1)},$$

for $i = 1, \ldots, J$. VNs send these probabilities to parity check factor nodes, which re-compute them according to the information received and the parity condition. Let $\hat{p}_{(0,1)}^{i \to j}$ be the message propagated from the ith variable node to the the jth check node, and let $p_{(0,1)}^{j \to i}$ be the message propagated from the jth check node to the the ith variable node. For instance, in the example shown in Figure 1.7, degree-3 CN1 receives $(\hat{p}_{(0)}^{1\to1}, \hat{p}_{(1)}^{1\to1}, \hat{p}_{(0)}^{2\to1}, \hat{p}_{(1)}^{2\to1})$ and it recomputes $(p_{(0)}^{1\to4}, p_{(1)}^{1\to4})$ as

follows:

$$\begin{aligned}(p_{(0)}^{1\to 4}, p_{(1)}^{1\to 4}) &= \mathrm{CHK}[\hat{p}_{(0)}^{1\to 1}, \hat{p}_{(1)}^{1\to 1}, \hat{p}_{(0)}^{2\to 1}, \hat{p}_{(1)}^{2\to 1}] \\ &= (\hat{p}_{(0)}^{1\to 1}\hat{p}_{(0)}^{2\to 1} + \hat{p}_{(1)}^{1\to 1}\hat{p}_{(1)}^{2\to 1}, \hat{p}_{(0)}^{1\to 1}\hat{p}_{(1)}^{2\to 1} + \hat{p}_{(1)}^{1\to 1}\hat{p}_{(0)}^{2\to 1}). \end{aligned} \tag{1.11}$$

where CHK refers to the update rules of the message propagated from variable nodes to check nodes. Note that the message received from x_4 is not used to re-compute $(p_{(0)}^{1\to 4}, p_{(1)}^{1\to 4})$, and that (1.11) already produces normalized probabilities. It proceeds similarly with the rest of messages.

Binary probability mas functions can be parametrized by a single value given that $\hat{p}_{(0)}^{i\to j} + \hat{p}_{(1)}^{i\to j} = 1, p_{(0)}^{j\to i} + p_{(1)}^{j\to i} = 1, i = 1, 2, \ldots, J, j = 1, 2, \ldots, K$. Given (1.11), following [40] we define that Likelihood ration (LR) and Log-likelihood ration (LLR) parametrizations as follows [40]

Likelihood Ratio (LR) :

$$\begin{aligned} Definition : \lambda_i &= \hat{p}_{(0)}^{i\to 1}/\hat{p}_{(1)}^{i\to 1} \\ (p_{(0)}^{1\to 4}, p_{(1)}^{1\to 4}) &= \mathrm{CHK}(\lambda_1, \lambda_2) = \frac{1+\lambda_1\lambda_2}{\lambda_1+\lambda_2}, \end{aligned} \tag{1.12}$$

Log-Likelihood Ratio (LLR) :

$$\begin{aligned} Definition : \Lambda_i &= \ln(\hat{p}_{(0)}^{i\to 1}/\hat{p}_{(1)}^{i\to 1}) \\ (p_{(0)}^{1\to 4}, p_{(1)}^{1\to 4}) &= \mathrm{CHK}(\Lambda_1, \Lambda_2) = 2\tanh^{-1}(\tanh(\Lambda_1/2)\tanh(\Lambda_2/2)), \end{aligned} \tag{1.13}$$

When the SPC nodes have degree larger than three, we can extend the CHK functions to more than two arguments via the relations [40]:

$$\mathrm{CHK}(x_1, x_2, \ldots, x_n) = \mathrm{CHK}(x_1, \mathrm{CHK}(x_2, \ldots, x_n)).$$

Considering parallel hardware implementation, we can also extend the CHK like following (i.e., when degree is four)

$$\mathrm{CHK}(x_1, x_2, \ldots, x_n) = \mathrm{CHK}(\mathrm{CHK}(x_1, x_2), \mathrm{CHK}(x_3, x_4)).$$

After VNs received the incoming messages from factor nodes, they re-compute the probabilities according to the information received [40]. For example, x_i receives $p_{(0)}^{1\to1}, p_{(1)}^{1\to1}, p_{(0)}^{2\to1}, p_{(1)}^{2\to1}$ and it recomputes normalized $(\hat{p}_{(0)}^{1\to3}, \hat{p}_{(1)}^{1\to3})$ output messages as follows

$$\begin{aligned}(\hat{p}_{(0)}^{1\to3}, \hat{p}_{(1)}^{1\to3}) &= \mathrm{VAR}[p_{(0)}^{1\to1}, p_{(1)}^{1\to1}, p_{(0)}^{2\to1}, p_{(1)}^{2\to1}] \\ &= \left(\frac{p_{10} p_{(0)}^{1\to1} p_{(0)}^{2\to1}}{p_{10} p_{(0)}^{1\to1} p_{(0)}^{2\to1} + p_{11} p_{(1)}^{1\to1} p_{(1)}^{2\to1}}, \frac{p_{11} p_{(1)}^{1\to1} p_{(1)}^{2\to1}}{p_{10} p_{(0)}^{1\to1} p_{(0)}^{2\to1} + p_{11} p_{(1)}^{1\to1} p_{(1)}^{2\to1}} \right).\end{aligned} \tag{1.14}$$

where VAR refers to the update rules of the message propagated from check nodes to variable nodes. Note that the message received from the factor node CN3 is not used, and the message received from the channel have to be considered.

Similar to the previous discussion, we have [40]

Likelihood Ratio (LR) :

$$\begin{aligned} Definition : \lambda_c &= p_{(0)}/p_{(1)} \\ \lambda_i &= p_{(0)}^{i\to1}/p_{(1)}^{i\to1} \\ (\hat{p}_{(0)}^{1\to3}, \hat{p}_{(1)}^{1\to3}) &= \mathrm{VAR}(\lambda_c, \lambda_1, \lambda_2) = \lambda_c \lambda_1 \lambda_2 \end{aligned} \tag{1.15}$$

Log-Likelihood Ratio (LLR) :

$$\begin{aligned} Definition : \Lambda_c &= \ln(p_{(0)}/p_{(1)}) \\ \Lambda_i &= \ln(p_{(0)}^{i\to1}/p_{(1)}^{i\to1}) \\ (\hat{p}_{(0)}^{1\to3}, \hat{p}_{(1)}^{1\to3}) &= \mathrm{VAR}(\Lambda_c, \Lambda_1, \Lambda_2) = \Lambda_c + \Lambda_1 + \Lambda_2 \end{aligned} \tag{1.16}$$

When the VNs have degree larger than three, we can extend the VAR functions to more than two arguments via the relations [40]:

$$\mathrm{VAR}(x_1, x_2, \ldots, x_n) = \mathrm{VAR}(x_1, \mathrm{VAR}(x_2, \ldots, x_n)).$$

we can also extend the VAR like following (i.e., when degree is four)

$$\mathrm{VAR}(x_1, x_2, \ldots, x_n) = \mathrm{VAR}(\mathrm{VAR}(x_1, x_2), \mathrm{VAR}(x_3, x_4)).$$

In computer programming, an overflow occurs when an arithmetic operation attempts to create a numeric value that is outside of the range that can be represented with a given number of digits. Using LLR, we avoid the overflow problem and this makes it more favorable in practical design. In LLR domain, for $x \gg 1$

$$\ln(\cosh(\mathrm{x}) \approx |x| - \ln(2)).$$

Thus, as shown in [40], a good approximation to the CHK function (1.13) is

$$\mathrm{CHK}(\lambda_1, \lambda_2) \approx |(\lambda_1 + \lambda_2)/2| - |(\lambda_1 - \lambda_2)/2| = \mathrm{sgn}(\lambda_1)\mathrm{sgn}(\lambda_2)\min(|\lambda_1|, |\lambda_2|). \tag{1.17}$$

This approximation yields a decoding algorithm of reduced complexity as the min-sum (MS) update rule. In practical hardware implementation, the MS update rule is widely used due to its low decoding complexity and because it allows a parallel hardware implementation.

Asymptotic analysis tools such as density evolution (DE) and extrinsic information transfer (EXIT) chart [86] can be used to predict the asymptotic performance of LDPC codes under the BP message-passing decoding. However, in the rest of the thesis, we focus on the BEC and the differential equation method, which is why we do not explain these tools in detail.

1.2.5 LDPC finite length performance

The performance of LDPC codes decreases as the block length becomes small. It is well known that the finite length performance of LDPC codes is determined by a set of structural properties of the Tanner graphs, such as *girth* and cycle distribution, and is limited by weaknesses of the iterative message passing decoding algorithm [16], such as weight distribution of pseudocodewords [94, 84, 99], and poor distance spectra. Specifically, the *girth* of a LDPC code is the length of the shortest cycle in the LDPC tanner graph, where a *cycle* starts from a node and ending at the same node, other nodes in the cycle is different. The PD algorithm often fails when there are cycles, e.g., cycle-6 in Figure 1.8, during the iterative message passing decoding. The performance of finite-length LDPC codes in the waterfall

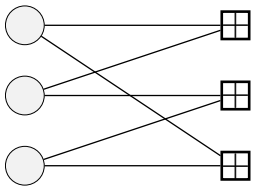
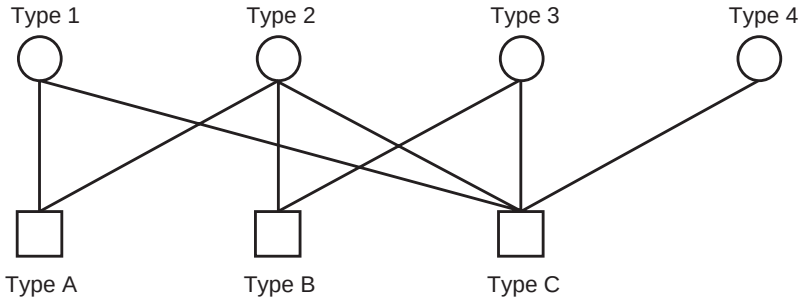


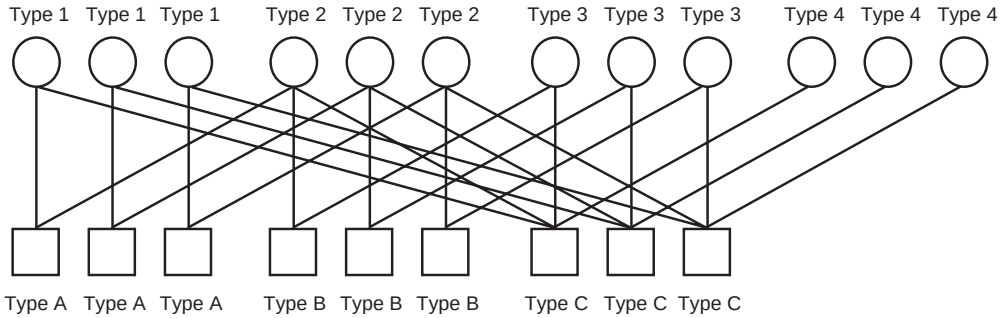
Figure 1.8: Tanner graph of a cycle-6.

region is provided in [71], a finite length analysis of LDPC codes under graph-cover decoding is studied in [100], and a finite length analysis of LDPC codes with large left degrees can be found in [109].

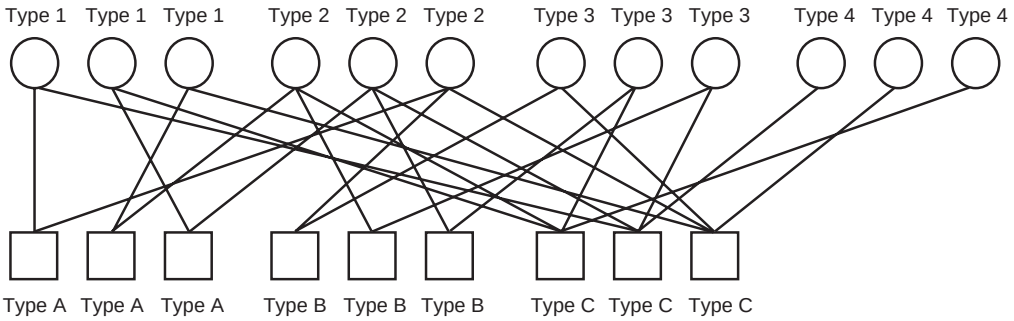
Many different methods are applied to design LDPC codes with large girth, and thus low error floor. In particular, for this thesis we are interested in protograph LDPC codes and quasi-cyclic (QC) LDPC codes [64]. A protograph or projected graph, is a Tanner graph with a relatively small number of nodes [62, 73, 96]. Here we use an example to illustrate how to construct a protograph-based LDPC code ensemble. The protograph shown in Figure 1.9 (a), it contains 4 variable nodes (named as type 1, 2, 3 and 4), and 3 check nodes, denoted as (A, B, C). To generate a code spanned by this protograph, we proceed by using a copy-and-permute process [96], as shown in Figure 1.9 (b) and Figure 1.9 (c). Specifically, in Figure 1.9 (b), the protograph has been copied three times, resulting in three disconnected subgraphs. The number of copies is referred to as the *Lifting Factor*. In Figure 1.9 (c), the endpoints of the three copies of each edge have been permuted at random among the three copies. After this swapping step, the three subgraphs are interconnected. In general, by increasing the lifting factor, the copy-and-permute operation can be applied to any protograph to obtain derived graphs of different sizes [96]. Suitably-designed protograph-based LDPC code ensembles have many desirable features, such as good iterative decoding thresholds and linear minimum distance growth, and they are asymptotically good [63]. In [64], the authors show empirically that the structural properties of the pre-lifted codes result in a decreased error floor and an improved minimum distance as the lifting factor increases.



(a) A simple protograph



(b) A protograph copied three times



(c) A generated LDPC code

Figure 1.9: In figure (a) we plot a simple protograph. In figure (b) we copy the protograph three times. In figure (c) we do the permutation step. (Figures borrowed from [96])

The algebraic structure of quasi-cyclic codes is determined by permutation matrices selected in the protograph-based construction that are restricted to be circulant. This class of codes allows simple encoding using shift registers, with a complexity that is linear in the block length [17]. Properly-designed QC graphs have been shown to perform as well as computer-generated random LDPC codes, regular or irregular, in terms of bit-error performance, block-error performance, and error floor for codes with short to moderate block lengths [45, 49]. More details of QC graphs are given in Chapter 3.

1.3 Generalized LDPC Codes

1.3.1 Background

As we discussed in the previous sections, LDPC codes are known to achieve channel capacity in the limit as the block length tends to infinity under sub-optimal BP decoding [98, 19]. However, it is unclear how to design LDPC codes able to approach the fundamental limits of information theory in the finite length case [79]. Generalized low-density parity-check (GLDPC) codes, which were first proposed by Tanner [95], have been shown to provide both good minimum distance and low decoding complexity [9]. In contrast to standard LDPC codes, which are represented by bipartite Tanner graphs where variable nodes and single parity-check (SPC) nodes are connected according to a given degree distribution (DD), in GLDPC codes the SPC nodes in the graph are replaced by generalized constraint (GC) nodes [95], as shown in Figure 3.1. The sub-code associated to each GC node is referred to as the component code. Examples of component codes used in the GLDPC literature are Hamming codes [44], Hadamard codes [107], Bose Chaudhuri Hocquenghem (BCH) code [61] or expurgated random codes [48, 22]. With powerful component codes, GLDPC codes have many potential advantages, including improved performance in noisy channels, fast convergence speed [68] and low error floor [48, 65].

Upon selecting a particular class of component codes, the DD of the GLDPC

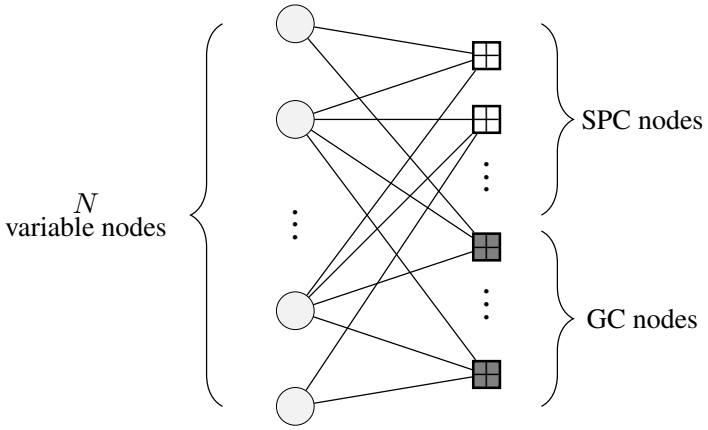


Figure 1.10: Tanner graph of a GLDPC code.

code ensemble can be optimized, and near-capacity iterative decoding thresholds can be achieved [44, 48, 1]. Capacity-achieving GLDPC code ensembles can also be obtained by spatially-coupling GLDPC block codes with regular DDs [43, 36]. Furthermore, the asymptotic exponents of the weight/stopping set spectrum for irregular and spatially-coupled GLDPC ensembles have been derived in [65] and [24], respectively. Based on these works, it is possible to design asymptotically good GLDPC code ensembles to achieve capacity-approaching iterative decoding thresholds and a minimum distance that grows linearly with the blocklength. To design GLDPC codes with large graph girth, many different techniques, such as quasi-cyclic designes, protograph-based design and random designs, have been proposed recently [49, 2, 60, 108]. A family of GLDPC codes for optical communications is proposed in [20], which uses Hamming, BCH, and Reed-Muller codes as GC nodes, and the Ashikhmin-Lytsin algorithm for decoding. Remarkable coding gains can be obtained from properly designed GLDPC codes, derived from multiple component codes. In [108], the authors consider imposing Hadamard code constraints at the check nodes for a low-rate approach, termed LDPC-Hadamard codes, and they also introduce a low-complexity message-passing based iterative soft-input soft-output (SISO) decoding algorithm. They further optimize the LDPC-Hadamard code ensemble by applying a low-complexity optimization technique based on approximating the density evolution by a one-dimensional dynamic system represented

by an extrinsic mutual EXIT chart. Simulation results show that the optimized LDPC-Hadamard codes not only offer better performance in the low-rate region than low-rate turbo-Hadamard codes, but also enjoy a fast convergence rate with respect to the block length. In [61], GLDPC codes with BCH or Reed-Solomon codes as component codes under bounded distance decoding are investigated. Simulation results show that the proposed technique yields competitive performance with a good decoding complexity trade-off for the BSC.

1.3.2 GLDPC decoding algorithms over the BEC

As in LDPC decoding, for the BEC, iterative decoding of GLDPC codes can be performed by means of *peeling decoding* (PD) algorithms [50, 57, 75], which iteratively remove from the Tanner graph variable nodes whose value is known. In the case of GLDPC codes, the derivation of the differential equations that predict the asymptotic performance requires to specify in advance the DD of the graph and a description of what kind of erasure patterns are locally decodable at any GC node, which depends on both the component codes and the corresponding decoding algorithm. In fact, the resulting decoding threshold of GLDPC codes heavily depends on this latter point [107, 22, 75]. For instance, for a $(2, 7)$ base DD in which all check nodes are $(7, 4)$-Hamming GC nodes, the asymptotic threshold over the BEC is $\epsilon^* \approx 0.7025$ if maximum likelihood (ML) decoding is performed at each GC node. However, it drops to $\epsilon^* \approx 0.5135$ if suboptimal bounded distance (BD) decoding is used instead of ML. In both cases, the coding rate is exactly the same. The reason for this difference in performance is that BD-decoded GC nodes only resolve erasure patterns up to degree $\mathtt{d} - 1$, where $\mathtt{d}$ is the minimum distance of the component code, whereas ML-decoded GC nodes can resolve a subset of erasure patterns of degree above $\mathtt{d} - 1$. Note, however, that this improvement of performance comes at the cost of higher complexity. Let K denote the blocklength of the component code. For the BEC, the ML-decoding complexity at GC nodes is of order $\mathcal{O}(K^3)$, since it is equivalent to solving a system of binary linear equations [10].

Algorithm 2 BD-PD

Remove from the Tanner graph of the GLDPC code all variable nodes with indexes in $\Gamma_{\boldsymbol{y}}$.

Construct Ψ, the index set of check nodes that correspond to either degree-one SPC nodes or GC nodes of degree less or equal to $\mathtt{d}-1$.

repeat

1) Select at random a member of Ψ.

2) Remove from the Tanner graph the check node with the index drawn in Step 1). Further, remove all connected variable nodes, and all attached edges.

3) Update Ψ.

until All variable nodes have been removed (successful decoding) or $\Psi = \emptyset$ (decoding failure).

Bounded Distance Peeling Decoding (BD-PD)

BD-PD is a suboptimal decoding method that considers decodable all GC nodes up to degree $\mathtt{d}-1$ [36, 59]. When BD is considered at GC nodes, a straightforward generalization of the PD algorithm presented in Section 1.2.3 is possible. Similar with the assumption in Section 1.2.3, we use a random GLDPC code of block length N to transmit over a BEC(ϵ). Decoding will be performed using a generalization of the PD algorithm [50] similar to that proposed for GLDPC codes in [75], denoted as PD with BD decoding at GC nodes (BD-PD). If we assume all GC nodes have minimum distance $\mathtt{d}$, the iterative algorithm BD-PD removes at random either a degree-1 SPC node or a GC node with degree smaller than $\mathtt{d}$. In Figure 1.11 we illustrate the BD-PD process assuming $\mathtt{d} = 3$. After the BEC transmission, BD-PD remove all correctly received VNs along with all adjacent edges from the tanner graph, as shown in Figure 1.11 (a). In the next stage, we pick at random a degree-1 SPC node or a GC nodes with degree smaller than $\mathtt{d}$, remove it from the graph, along with the adjacent edge, as shown in Figure 1.11 (b). Furthermore, the VN connected with this edge is decodable and will be removed from the graph, along with all adjacent edges, as shown in Figure 1.11 (c). Continuously, the BD-

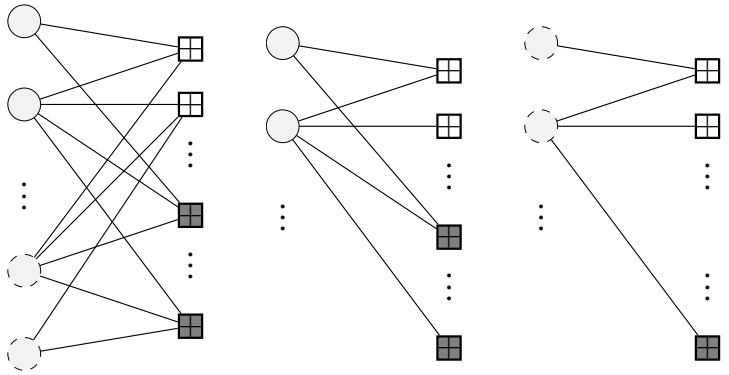


Figure 1.11: The bounded distance decoding process.

PD algorithm repeat this process until there are no VN left in the graph, which corresponding to a decoding *success*. If there are no more degree-1 SPC or GC nodes with degree smaller than `d` before successful decoding, we shall say that there is a decoding *failure*. Algorithm 2 summarizes the BD-PD algorithm. The threshold of BD-PD can be analyzed by extending the differential equation method proposed in [50, 75].

Beyond BD-PD

While deriving the asymptotic differential equations to analyze BD-PD follows a straightforward extension of the standard PD differential equations for LDPC codes [50], the GLDPC asymptotic analysis of PD under ML-decoded component codes (ML-PD) requires the use of multi-edge-type (MET) DDs [86] to track down all possible decodable erasure patterns at GC nodes [43, 75]. As a consequence, the list of code parameters to jointly optimize becomes cumbersome. Specifically, the parameters include the description of the multi-edge type DD, the position of GC nodes in the graph, the edge labelling at every GC node used to determine positions in the component block code, and the list of locally ML-decodable erasure patterns. In [22], the authors were able to incorporate ML-decoded GC nodes without resorting to multi-edge type DDs by analyzing the GLDPC average performance using extrinsic information (EXIT) charts when each GC node

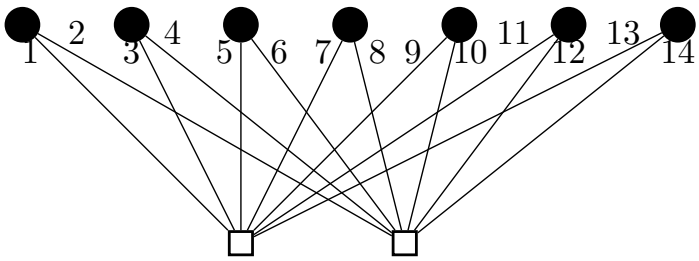


Figure 1.12: Tanner graph corresponding to a $(2, 7)$-regular protograph base matrix. We include the edge labeling used to define the ensemble degree distribution. (Figure is borrowed from [75])

in the graph is selected at random within the family of block component codes with fixed block length and minimum distance larger than 2. This approach has a design caveat though, as it does neither allow the use of a single type of component codes, nor to narrow down the family of component codes by fixing the minimum distance.

As an example of a class of MET DD required to fully specify ML-decoded GC nodes, we present here the protograph-based GLDPC code ensemble described in [75]. In this paper, to define the DD of the proposed codes, the authors labeled each edge in the base matrix $\mathbf{B}$ connecting a different pair of nodes. For example, Figure 1.12 shows the Tanner graph associated with the base matrix of a $(2, 7)$-regular GLDPC code. Edges are labeled from 1 to 14. Consequently, the degree of generalized codes is specified by 14-length vectors. In the original graph, only two possible degrees exist:

$$c_1 = [1\ 0\ 1\ 0\ 1\ 0\ 1\ 0\ 1\ 0\ 1\ 0\ 1\ 0]$$

$$c_2 = [0\ 1\ 0\ 1\ 0\ 1\ 0\ 1\ 0\ 1\ 0\ 1\ 0\ 1]$$

However, after PD intialization, in which variable nodes are removed from the graph, the extended set of GC degrees present in the residual graph is 256. In general, the more structure we include in the GLDPC graph, the more dimension the MET DD has, which dramatically impacts the complexity of the GLDPC code analysis. In this thesis, we propose a new class of GLDPC code ensembles and

new analysis tools that do not require the use of MET DDs and provides a simple design framework from which powerful finite-length GLDPC codes are proposed.

1.3.3 GLDPC decoding algorithms over general channels

Compared to the conventional belief propagation update rules for LDPC decoders, the only difference of the iterative message passing of GLDPC codes is how to process probabilistic messages at the GC nodes. In this regard, the processing depends on the codebook of the chosen component code. We take the $(2, 6)$-regular GLDPC code as a running example, where the component code used at GC nodes is a shortened $(6, 3)$ Hamming code, whose codebook can be expressed in matrix form as

$$\mathbf{C}^{(6,3)} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 \\ 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 \end{bmatrix}. \tag{1.18}$$

The GLDPC update rule at GC nodes is determined by the component codebook $\mathbf{C}^{(6,3)}$. Let Λ_j denote the input LLR message coming from the j-th variable node connected to the GC node, where index j, $j = 1, 2, \ldots, 6$, corresponds to the jth input to the component code. Let $\tilde{\Lambda}_j$ denote the output LLR message to be sent to the j-th variable node. In Appendix D.1, we show that $\tilde{\Lambda}_j$, $j = 1, 2, \ldots, 6$, can be computed as follows

$$\begin{aligned} \tilde{\Lambda}_j = &\log \left[\sum_{\substack{i \in \{1,8\} \\ C_{i,j}=0}} \exp \left(\sum_{\substack{m \in \{1,6\} \\ m \neq j}} \mathbb{I}[(\mathbf{C}_{i,m}^{(6,3)} = 0)](\Lambda_{pj} - \Lambda^*) \right) \right] \\ &- \log \left[\sum_{\substack{i \in \{1,8\} \\ C_{i,j}=1}} \exp \left(\sum_{\substack{m \in \{1,6\} \\ m \neq j}} \mathbb{I}[(\mathbf{C}_{i,m}^{(6,3)} = 1)](\Lambda_{pj} - \Lambda^*) \right) \right], \end{aligned} \tag{1.19}$$

where $\mathbf{C}_{i,m}^{(6,3)}$ denotes the m-th bit of the i-th codeword, $i = 1, 2, \ldots, 8$, $\Lambda^* = \max_j \Lambda_j$, and we use the *log-sum-exp trick* to avoid numerical issues in the evaluation of the exponential terms.[1] The value of Λ^* can be efficiently computed using a digital comparator. Note that the decoding complexity at GC nodes is determined by the degree of the GC nodes and grows exponentially as the degree increases.

1.3.4 Contributions of the proposed asymptotic analysis technique in this work

In Chapter 2 of this thesis, we propose a flexible and efficient asymptotic analysis of GLDPC codes over the BEC that allows us to easily incorporate ML-decoded GC nodes with specific properties and still maintain a random definition of the graph degree distribution. In other words, there is no need for multi-edge type DD to define the GLDPC ensemble, which opens the door for flexible code design and optimization, as the number of parameters that define both the ensemble and decoding algorithm remain small. An example of such novel design methodology are the class of GLDPC code ensembles analyzed in Chapter 2. Instead of selecting a particular class of component codes and optimizing the graph DD, we are interested in analyzing the trade-off between coding rate and iterative decoding threshold of GLDPC code ensembles with fixed DD as we increase the fraction of GC nodes in the graph. We study the trade-off between iterative decoding threshold, coding rate and minimum distance, and we study what is the required fraction of GC nodes required to minimize the gap to channel capacity and still provides a code ensemble with linear growth of the minimum distance w.r.t. the block length. These results have been summarized in the following publications:

- Yanfang Liu, Pablo M. Olmos, and Tobias Koch. A Probabilistic Peeling Decoder to Efficiently Analyze Generalized LDPC codes over the BEC. Sub-

[1] The *log-sum-exp trick* works as follows: let $\mathbf{a} = [a_1, a_2, \ldots, a_d]$ be a real-valued vector. Instead of directly evaluating $b = \log(\sum_{i=1}^{d} \exp(a_d))$, we first compute $a^* = \max_i a_i$ and then we compute b as $b = a^* + \log(\sum_{i=1}^{d} \exp(a_d - a^*))$.

mitted to IEEE Transactions on Information Theory (2nd review, available: `https://arxiv.org/pdf/1709.00873.pdf`).

- Yanfang Liu, Pablo M. Olmos and Tobias Koch On LDPC code ensembles with generalized constraints. Proceedings of 2017 IEEE Information Theory (ISIT), 2017 IEEE pp. 371-375, Acchen, Germany, June, 2017.

The obtained results demonstrate that simple regular GLDPC codes are indeed robust if the right fraction of GC nodes is included in the graph. In Chapter 3, we derive practical implementations of such codes using quasi-cyclic graphs and demonstrate their application for URLLC. We further study low-complexity decoding schemes with quantization and sub-optimal min-sum like update rules. Our results demonstrate that we can achieve remarkable gains compared to existing schemes in the literature at similar complexity. These results have been summarized in the following publications:

- Yanfang Liu, Pablo M. Olmos, and David G. M. Mitchell. On Generalized LDPC Codes for 5G Ultra Reliable Communication. Proceedings of the 2018 IEEE Information Theory Workshop (ITW), Guangzhou, China, November, 2018.

- Yanfang Liu, Pablo M. Olmos, and David G. M. Mitchell. Generalized LDPC Codes for Ultra Reliable Low Latency Communication in 5G and Beyond. Accepted for publication in IEEE Access, Special issue on Advances in Channel Coding in 5G and Beyond. November 2018.

2

Probabilistic Peeling Decoder Analysis of GLPDC codes

In this chapter, we propose an analysis methodology that allows to easily incorporate into the PD algorithm ML-decoded GC nodes with specific properties, such as particular value of the minimum distance $\mathtt{d}$ or how many erasure patterns beyond minimum distance it can decode. We develop a probabilistic description of all components of the GLDPC code, namely the graph degree distribution (DD), the presence of GC nodes in the graph, and the decoding method implemented at GC nodes. Regarding the latter aspect, we parameterize the decoding capabilities of at every node with a blocklength-K component code by a vector $(p_1, p_2, ..., p_K)$, where $p_w \in [0, 1]$, $w \in \{1, \ldots, K\}$, is the probability that a weight-w erasure pattern chosen at random is decodable. Thus, p_w is the fraction of decodable weight-w erasure patterns. Note that if we take $p_w = 1$ for $w \leq \mathtt{d} - 1$ and $p_w = 0$

for $w = \{\mathtt{d}, \ldots, K\}$, we recover BD-PD. We show how to properly incorporate such a probabilistic description of component codes into the PD algorithm, and denote the resulting algorithm as *probabilistic PD* (P-PD). Due to its probabilistic nature, the asymptotic analysis of P-PD does not require the use of multi-edge type DDs. We show by computer simulations that the P-PD performance accurately predicts the actual GLDPC performance when ML decoding is performed at GC nodes. We note that the proposed techniques are valid for binary GLDPC codes and that we do not consider non-binary LDPC codes [52], which can also be considered a special class of GLDPC codes.

The performance predicted using P-PD is valid for any linear component code of blocklength-K and decoding profile $(p_1, p_2, ..., p_K)$. To analyze a family of linear component codes of blocklength-K and minimum distance $\mathtt{d}$, we employ two bounds to compute the GLDPC coding rate. The Hamming or sphere-packing bound [56] is used to determine a converse bound on the rate of the GLDPC code ensemble as a function of a triplet of $(\nu, \mathtt{d}, K)$. The Varshamov bound is considered to determine an achievable rate of the GLDPC code ensemble [34]. In many scenarios of interest, we show that these bounds are sufficiently tight and thus relevant for the code designer.

By employing a probabilistic description of the decoding capabilities at GC nodes, we are able to analyze a large class of GLDPC code ensembles and beyond-BD decoding methods with a fairly small set of parameters. We demonstrate our approach by analyzing the tradeoff between coding rate and iterative decoding threshold of GLDPC code ensembles with fixed DD, referred to as the base DD, as we increase the fraction ν of GC nodes in the graph. This approach is novel in the literature and we believe it is appealing from a design perspective, since one might be interested in introducing a certain amount of GC nodes in the Tanner graph of a given LDPC code, aiming at reducing the gap to channel capacity at the resulting coding rate, and at the same time improving the minimum distance of the code and thus the error floor.

We illustrate our analysis for both regular GLDPC code ensembles using $(2, 6)$,

$(2, 7)$, $(2, 8)$ and $(2, 15)$ base DDs and irregular GLDPC code ensembles with similar graph densities [77, 31]. To obtain realistic values for the coding capabilities of the component codes, we have performed an exhaustive search of linear block codes of lengths $r \in [6, 7, 8, 15]$, including Hamming codes, Cyclic codes, Quasi Cyclic codes and Cordaro-Wagner Codes, and tabulated their corresponding description in terms of minimum distance $\mathtt{d}$ and $(p_1, p_2, ..., p_K)$. In all cases, we show that a large fraction of GC nodes is required in the GLDPC graph to reduce the original gap to capacity. However, the closest gap to capacity is not achieved at $\nu = 1$, but a smaller value must be used. Namely, there exists a critical ν^* value for which the gap to capacity is minimum. Furthermore, the best results are obtained for high-rate component codes, suggesting that the use of very powerful component codes does not pay off, since the gain in threshold does not compensate for the severe decrease of the GLDPC code rate. We also include into our analysis the weight spectral analysis of GLDPC ensembles in [24] to explore the range of ν values for which the GLDPC ensembles reduce the original gap to capacity and at the same time maintain a linear growth of the minimum distance with the block length.

Finally, we illustrate how to incorporate further design techniques that can help to reduce the gap to capacity of the code ensembles. Specifically, we discuss both random puncturing [66] and a simple class of doubly generalized LDPC (DG-LDPC) codes [101, 102]. In general, the methodology presented in this work is flexible and decouples the problems of bounding the GLDPC coding rate and the asymptotic analysis of the ensemble. In this regard, broader classes of component codes at variable nodes and GC nodes could also be incorporated in a systematic way.

The chapter is organized as follows. In Section 2.1, we introduce GLDPC code ensembles and the notation used to characterize the DDs. Sections 2.2 and 2.3 present the decoding algorithm and its asymptotic analysis. In Section 2.4 we bound the GLDPC code rate and analyze the rate-threshold tradeoff as a function of the fraction ν of GC nodes in the graph. The behavior of the GLDPC

code ensembles with specific component codes is analyzed in Section 2.5. Finally, Sections 2.6 and 2.7 consider techniques to improve the asymptotic behavior of the code ensemble, by means of random puncturing and generalized variable nodes. In Section 2.8 we conclude the chapter with a discussion of our results.

2.1 GLDPC ensembles with increasing fraction of GC nodes

In this section, we introduce the GLDPC code ensembles that will be analyzed in the rest of the chapter and the notation used to define their DD.

2.1.1 Degree distribution

As illustrated in Fig. 2.1, the Tanner graph of every member in the ensemble contains `n` variable nodes (coded bits) and `c` parity-check nodes, among which a fraction ν corresponds to GC nodes while the rest corresponds to SPC nodes. We denote by `E` the number of edges in the Tanner graph and we define the degree of a node as the number of edges connected to it.

The DD of the ensemble is characterized as follows. The vector $\overline{\lambda} = (\lambda_1, \lambda_2, ..., \lambda_J)$ is the *left* DD, where λ_i represents the fraction of edges (w.r.t. `E`) connected to a variable node of degree i. Given $\overline{\lambda}$, N and `E` are related by [86]

$$N = \mathtt{E} \sum_{i=1}^{J} \lambda_i / i. \tag{2.1}$$

The *right* DD is defined by two vectors $\overline{\rho}_p = (\rho_{p1}, \rho_{p2}, ..., \rho_{pK})$ and $\overline{\rho}_c = (\rho_{c1}, \rho_{c2}, ..., \rho_{cK})$, where ρ_{pj} denotes the fraction of edges (w.r.t. `E`) connected to a SPC node that has degree j and ρ_{cj} denotes the fraction of edges (w.r.t. `E`) connected to a GC node that has degree j. Throughout the paper, we use the subscript p for any DD component related to standard *parity* check nodes and the subscript c for any DD component related to generalized *component* codes. The DD is then characterized by the tuple $(\overline{\lambda}, \overline{\rho}_p, \overline{\rho}_c, \nu)$ and the ensemble of codes generated by this DD is denoted by $\mathcal{C}_{\overline{\lambda}, \overline{\rho}_p, \overline{\rho}_c, \nu}$. Since the fraction of GC nodes in

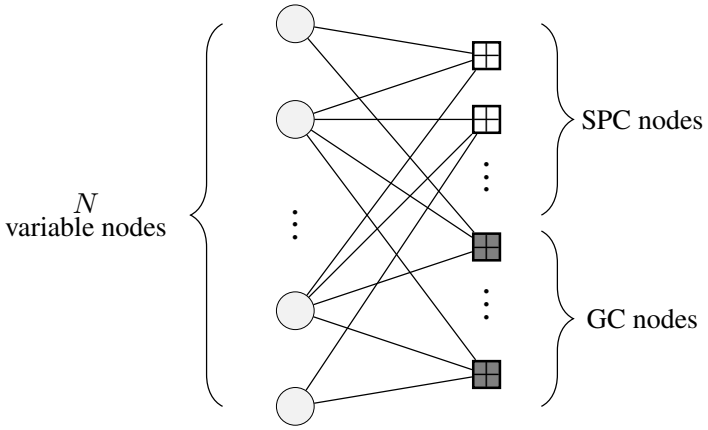


Figure 2.1: Tanner graph of a GLDPC code.

the graph is ν, the following must hold:

$$\nu = \frac{\sum_{j=1}^{K} \rho_{cj}/j}{\sum_{u=1}^{K} (\rho_{cu} + \rho_{pu})/u}. \tag{2.2}$$

For simplicity, we restrict most of our analysis to the class of GLDPC ensembles characterized by variable nodes with constant degree J and SPC and GC nodes with constant degree and K. The Tanner graph of any code in this ensemble contains N variable nodes, $\mathtt{E} = JN$ edges, $\nu \frac{J}{K} N$ GC nodes, and $(1-\nu)\frac{J}{K}N$ SPC nodes. The DD of the GLDPC codes is characterized by the triple (J, K, ν), and the ensemble of codes generated by this DD is denoted by $\mathcal{C}_{J,K,\nu}$. The DD of the LDPC ensemble obtained by taking $\nu = 0$ is defined as the *base DD*, and the corresponding LDPC code ensemble is referred to as the *base ensemble*. The coding rate of the base ensemble is denoted by $\mathtt{R}_0$ and can be computed as:

$$\mathtt{R}_0 = 1 - \frac{J}{K}. \tag{2.3}$$

Finally, we assume that the incoming edges to every degree-K GC node are assigned uniformly at random to each position of the component code.

2.1.2 The coding rate of the $\mathcal{C}_{J,K,\nu}$ ensemble

As discussed in the introduction of the paper, we propose tools to analyze the decoding performance of GLDPC under ML-decoded GC nodes that do not require

to set in advance a specific component code to be used as the GC nodes. Instead, we consider the family of linear block codes with blocklength K and minimum distance $\mathtt{d}$, and we use the classical results on linear block codes to bound the coding rate of the GLDPC code ensembles.

Let $\mathtt{k}^{(\ell)} \in \mathbb{N}^+$, $\ell = 1, \ldots, \nu\mathtt{E}/K$, be the number of rows in the parity-check matrix associated with the component code of the ℓ-th GC node.

Lemma 1. The design rate $\mathtt{R}(\nu)$ of the $\mathcal{C}_{J,K,\nu}$ ensemble is

$$\mathtt{R}(\nu) = \mathtt{R}_0 - \nu(1-\mathtt{R}_0)(\mathtt{k}_{\text{avg}} - 1), \tag{2.4}$$

where $\mathtt{k}_{\text{avg}} \triangleq (\nu\frac{\mathtt{E}}{K})^{-1} \sum_{\ell=1}^{\nu\frac{\mathtt{E}}{K}} \mathtt{k}^{(\ell)}$ is the average number of rows in the parity-check matrix of the component codes.

Proof. Any SPC node in the Tanner graph accounts for a single row in the parity-check matrix of the GLDPC code, and any GC node accounts for $\mathtt{k}^{(\ell)}$ rows. Thus, the design rate $\mathtt{R}(\nu)$ is given by

$$\begin{aligned}\mathtt{R}(\nu) &= 1 - \frac{(1-\nu)\frac{\mathtt{E}}{K} + \sum_{\ell=1}^{\nu\frac{\mathtt{E}}{K}} \mathtt{k}^{(\ell)}}{N} = 1 - \frac{(1-\nu)\frac{\mathtt{E}}{K} + \nu\frac{\mathtt{E}}{K}\mathtt{k}_{\text{avg}}}{\mathtt{E}/J} \\ &= \mathtt{R}_0 - \nu(1-\mathtt{R}_0)(\mathtt{k}_{\text{avg}} - 1). \end{aligned} \tag{2.5}$$

□

Note that the second term in (2.4) accounts for the rate loss at GC nodes. When the component codes are linear block codes with minimum distance $\mathtt{d}$, we obtain the following bounds on $\mathtt{R}(\nu)$:

Lemma 2. If all component codes in the $\mathcal{C}_{J,K,\nu}$ ensemble are linear block codes with minimum distance $\mathtt{d} > 2$, then

$$\mathtt{R}(\nu) \leq \mathtt{R}_0 - \nu(1-\mathtt{R}_0)\log_2\left(\frac{1}{2}\sum_{q=0}^{\lfloor\frac{\mathtt{d}-1}{2}\rfloor}\binom{K}{q}\right). \tag{2.6}$$

Furthermore, there exists a set of linear block codes to be used as component codes such that

$$\mathtt{R}(\nu) \geq \mathtt{R}_0 - \nu(1-\mathtt{R}_0)\left\lceil\log_2\left(\frac{1}{2} + \frac{1}{2}\sum_{q=0}^{\mathtt{d}-2}\binom{K-1}{q}\right)\right\rceil. \tag{2.7}$$

Here, we use $\lceil\cdot\rceil$ and $\lfloor\cdot\rfloor$ to denote the ceiling and floor functions, respectively. The two bounds coincide, for example, when $\mathtt{d} = 3$ and $K = 2^z - 1$, where $z \in \mathbb{Z}_+$.

Proof. First, the condition $\mathtt{d} > 2$ is required to differentiate between the rate loss at SPC nodes, which are block codes with minimum distance 2, and at GC nodes. We start by proving the converse bound in (2.6). By the sphere-packing bound [15, Theorem 12, p.531], any component code with blocklength K and minimum distance $\mathtt{d}$ must satisfy

$$2^{K-\mathtt{k}} \leq \frac{2^K}{\sum_{q=0}^{\lfloor \frac{\mathtt{d}-1}{2} \rfloor} \binom{K}{q}}, \tag{2.8}$$

where $\mathtt{k}$ is the number of rows in the parity-check matrix. Here we consider non redundant parity check matrices (i.e. $K - \mathtt{k}$ is exactly the information dimension of the code). This implies that the term $(\mathtt{k}_{\text{avg}} - 1)$ in (2.4) is bounded by

$$\mathtt{k}_{\text{avg}} - 1 \geq \log_2 \left(\frac{1}{2} \sum_{q=0}^{\lfloor \frac{\mathtt{d}-1}{2} \rfloor} \binom{K}{q} \right), \tag{2.9}$$

which proves (2.6). Regarding the achievable bound in (2.7), the Varshamov Bound [34, Theorem 2.9.3] guarantees the existence of a linear component code with blocklength K and minimum distance at least $\mathtt{d}$ if

$$2^{K-\mathtt{k}} \geq 2^{K - \left\lceil \log_2 \left(1 + \sum_{q=0}^{\mathtt{d}-2} \binom{K-1}{q} \right) \right\rceil}. \tag{2.10}$$

If the above condition is satisfied, then there exists a set of linear block codes to be used as component codes with blocklength K and minimum distance at least $\mathtt{d}$ such that

$$\mathtt{k}_{\text{avg}} - 1 \leq \left\lceil \log_2 \left(\frac{1}{2} + \frac{1}{2} \sum_{q=0}^{\mathtt{d}-2} \binom{K-1}{q} \right) \right\rceil, \tag{2.11}$$

which proves (2.7).

Finally, if we substitute $\mathtt{d} = 3$ and $K = 2^z - 1$ for some $z \in \mathbb{Z}_+$ into (2.6) and (2.7), a straightforward computation shows that the converse bound in (2.6) can be simplified to

$$\mathtt{R}(\nu) \leq \mathtt{R}_0 - \nu(1 - \mathtt{R}_0)(z - 1), \tag{2.12}$$

and, likewise, the achievable bound in (2.7) simplifies to

$$\mathtt{R}(\nu) \geq \mathtt{R}_0 - \nu(1-\mathtt{R}_0)(z-1). \tag{2.13}$$

□

2.1.3 Growth rate of the weight distribution of the $\mathcal{C}_{J,K,\nu}$ ensemble

A useful tool for analysis and design of LDPC codes and their generalizations is the asymptotic exponent of the weight distribution. The growth rate of the weight distribution was introduced in [26] to show that the minimum distance of a randomly-generated regular LDPC code with variable nodes of degree of at least three is a linear function of the codeword length with high probability. The growth rate of the weight distribution for a class of doubly generalized LDPC (D-GLDPC) codes was introduced in [24]. The $\mathcal{C}_{J,K,\nu}$ GLDPC code ensemble can be seen as a particular instance of the codes analyzed in that work. The *weight spectral shape* of the $\mathcal{C}_{J,K,\nu}$ ensemble captures the behavior of codewords whose weight is linear in the block length N and is defined by

$$G(\alpha) \triangleq \lim_{N\to\infty} \frac{1}{N} \log \mathbb{E}_{\mathcal{C}_{J,K,\nu}}[X_{\alpha N}] \tag{2.14}$$

for $\alpha > 0$, where X_w denotes the number of codewords of weight-w of a randomly chosen code in the $\mathcal{C}_{J,K,\nu}$ code ensemble. This limit assumes the inclusion of only those positive integers for which $\alpha N \in \mathbb{Z}$. We define the *critical exponent codeword weight ratio* as $\hat{\alpha} \triangleq \inf\{\alpha \geq 0 | G(\alpha) \geq 0\}$. If $\hat{\alpha} > 0$, then the code's minimum distance asymptotically grows as $\mathcal{O}(\hat{\alpha}N)$ and the ensemble is said to have good growth rate behavior. If $\hat{\alpha} = 0$, then the minimum distance of the code may still grow with the block length N but at a slower rate, e.g., as $\mathcal{O}(\log(N))$.

Lemma 3. If all component codes in the $\mathcal{C}_{J,K,\nu}$ ensemble are linear block codes with minimum distance $\mathtt{d} > 2$, then $\hat{\alpha} > 0$ for $J > 2$. For $J = 2$, $\hat{\alpha} > 0$ if and only if

$$\nu > \frac{K-2}{K-1} \triangleq \hat{\nu}. \tag{2.15}$$

Otherwise, $\hat{\alpha} = 0$.

Proof. The lemma follows directly by particularizing the results in [24] [Section II] to the $\mathcal{C}_{J,K,\nu}$ ensemble. □

2.2 Probabilistic Peeling Decoding over the BEC

Suppose we use a random sample of the $\mathcal{C}_{J,K,\nu}$ ensemble to transmit over a BEC(ϵ). For this channel, each of the N coded bits is erased with probability ϵ. Without loss of generality, we assume that the all-zero codeword is transmitted, hence the received vector $\boldsymbol{y}$ belongs to the set $\{0,?\}^N$, where ? denotes an erasure. Let $\Gamma_{\boldsymbol{y}} \subseteq \{1,\ldots,N\}$ be the index set of the bits correctly received, namely $y_i = 0$ for all $i \in \Gamma_{\boldsymbol{y}}$. Decoding will be performed using a generalization of the PD algorithm [50] similar to that proposed for GLDPC codes in [75]. The final formulation of the decoding algorithm depends on the decoding capabilities we assume at GC nodes. For instance, if we assume BD decoding at component codes, then the generalized PD algorithm, denoted as BD-PD, proceeds as described in Algorithm 2 (See Section 1.3.2).

BD-PD is a suboptimal decoding method that considers decodable all GC nodes up to degree $\mathtt{d}-1$ [36, 59]. However, it ignores the fact that any component code will be able to decode a certain fraction of erasure patterns of weight equal to or greater than $\mathtt{d}$. As already reported in various works, e.g., [43, 75], the GLPDC code performance dramatically improves if we consider ML decoding at GC nodes. In principle, to consider ML decoding at GC nodes, we have to specify a full list of decodable erasure patterns and, label each of the incoming edges at every GC node to differentiate between decodable and non-decodable GC nodes. As shown in [75], incorporating this labelling into the asymptotic analysis requires the use of multi-edge type DDs.

In order to incorporate beyond-BD decoding at GC nodes into our analysis, and at the same time maintain a formulation compatible with the random definition of the $\mathcal{C}_{J,K,\nu}$ ensemble, we will further constrain the family of component codes to be used at degree-K GC nodes. More specifically, we assume that the fraction of ML-decodable weight-w erasure patterns at every GC node is given by some

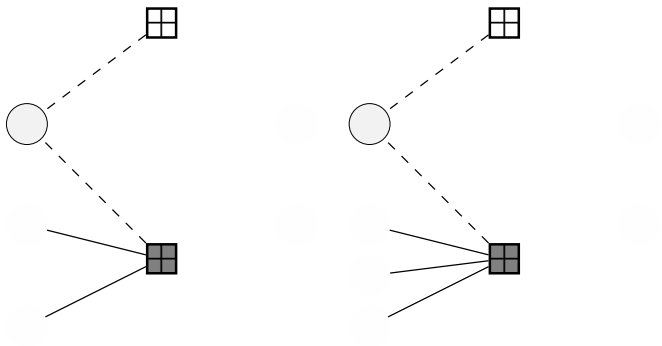


Figure 2.2: We illustrate one iteration of the P-PD algorithm. Assuming GC nodes with $\mathtt{d} = 3$, in (a) right after dashed edges are removed, the remaining GC node (gray shadowed) becomes degree-2 and thus it will be considered decodable in future iterations. In (b), after the GC node becomes degree-3, a sample from Bernoulli Random Variable with success probability equal to p_3 is drawn. If the sample is a success, we tagged the GC node as decodable for future iterations. Otherwise, it is tagged as non-decodable and only after the node losses any additional edge the tag can be reverted to decodable.

$p_w \in [0, 1], w = 1, \ldots, K$. Thus, the family of component codes under analysis is the family of blocklength-K linear block codes with minimum distance $\mathtt{d}$ and with decoding profile described by the vector $\mathbf{p} = (p_1, \ldots, p_K)$. Note that if the minimum distance of the component code is $\mathtt{d}$, then $p_w = 1$ for $w \leq \mathtt{d} - 1$. The bounds on $\mathtt{R}(\nu)$, predicted in Lemma 2, could in principle be refined according to $\mathbf{p}$. While this is an interesting open question, we will later show that the bounds are tight in certain scenarios and there is little room for refinement.

By exploiting the fact that incoming edges at every GC node are assigned to each position of the component code uniformly at random, we can incorporate ML-decoded GC nodes into the PD as shown in Algorithm 3, denoted as *probabilistic PD (P-PD)*. Observe that the key P-PD feature is to tag GC check nodes as decodable with probabilities given by $\mathbf{p}$ only when they lose one or more edges, which may happen either at the initialization or after a connected variable is removed. If only one decodable check node is removed per iteration, after every P-PD iteration only a few GC nodes can change its state (from non-decodable to

Algorithm 3 P-PD

Remove from the Tanner graph of the GLDPC code all variable nodes with indexes in $\Gamma_{\boldsymbol{y}}$.

for all GC nodes **do**

If the GC has degree w, tag the check node as *decodable* with probability p_w.

end for

Construct Ψ, the index set of check nodes corresponding to either degree-one SPC nodes or GC nodes tagged as *decodable*.

repeat

1) Select at random a member of Ψ.

2) Remove from the Tanner graph the check node with the index drawn in Step 1). Further remove all connected variable nodes and all attached edges.

3)

for every non-decodable GC node that has lost one or more edges in the current iteration **do**

If the GC has degree w, draw a sample of a Bernoulli distribution with success probability p_w. If the sample is a success, tag the check node as *decodable*.

end for

4) Update Ψ.

until All variable nodes have been removed (successful decoding) or $\Psi = \emptyset$ (decoding failure).

decodable). See Fig. 2.2 for an explanatory diagram. Thus, at every iteration, P-PD emulates the ML decoding operation of a degree-w GC node by drawing the decoding capability according to a Bernoulli distribution with parameter p_w, $w \in \{1, \ldots, K\}$. Note that P-PD is a procedure that allows for simpler analysis rather than a practical decoding algorithm. Further note that we recover the bounded distance PD (BD-PD) algorithm from P-PD if we set $p_w = 0$ for $w \geq \mathtt{d}$

and $p_w = 1$ otherwise.

2.2.1 Comparing the P-PD and ML-PD performances by Monte Carlo simulation

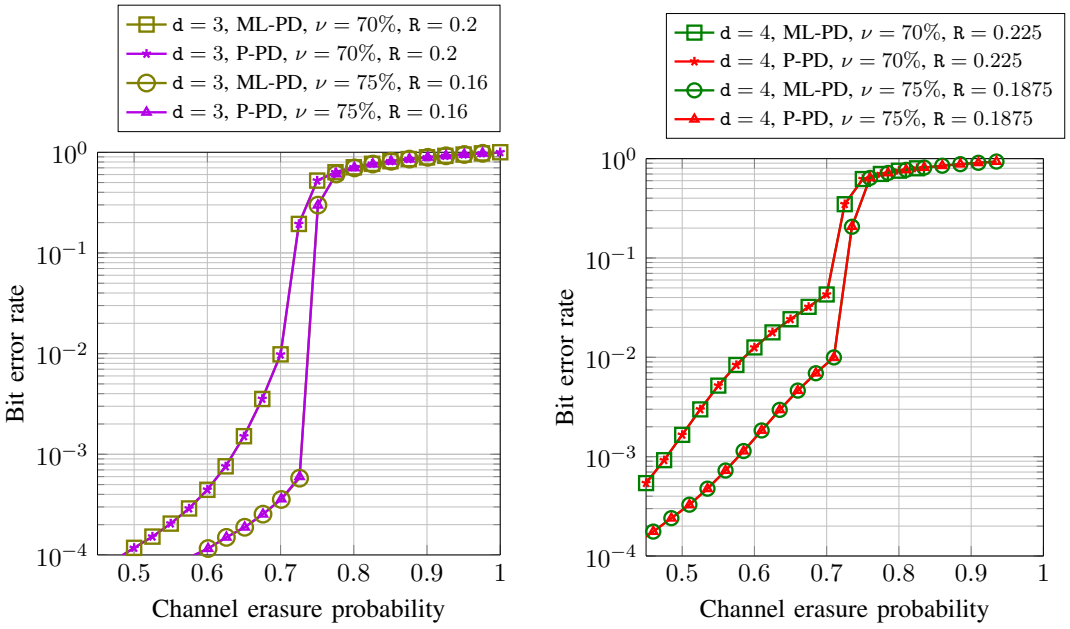
If we select a specific component code, we can compare the simulation performance of the $\mathcal{C}_{J,K,\nu}$ ensemble for the corresponding parameters under P-PD with that of the practical GLDPC codes with GC nodes that are decoded via ML, using the actual parity-check matrix of the component codes. We refer to this latter case as ML-PD.

More precisely, for a given finite blocklength N, fixed $\nu \in [0, 1]$, and base DD, we generate a member of the $\mathcal{C}_{J,K,\nu}$ ensemble as follows:

1. Generate at random a Tanner graph according to the (J, K) base DD. Then, select at random a fraction ν of check nodes to be used as GC nodes. Overall, the graph contains N variable nodes, $\nu\mathtt{E}/K$ GC nodes and $(1-\nu)\mathtt{E}/K$ SPC nodes.

2. For each of the $\nu\mathtt{E}/K$ GC nodes, we generate uniformly at random a permutation of the set $\{1, 2, \ldots, K\}$, which is used to associate each of the incoming edges to the GC node to a position in the component code.

We estimate by Monte Carlo simulation the bit error rate (BER) over the BEC achieved by both P-PD, which follows Algorithm 3, and ML-PD, which uses a look-up table of decodable erasure patterns. In Fig. 2.3 (a), we plot the BER as a function of the channel erasure probability of P-PD and ML-PD for a $(2, 6)$-regular base DD with a rate-1/2 Hamming $(6, 3)$ linear block code as component code. In Fig. 2.3 (b), we plot the same quantities for a $(2, 8)$-regular base DD using a rate-1/2 $(8, 4)$ Hamming component code. Results have been averaged over 10 generated samples from the $\mathcal{C}_{J,K,\nu}$ ensemble. Observe the perfect match between the BERs for P-PD and ML-PD in all cases. This illustrates that we are not sacrificing accuracy with the probabilistic description of the decoder, as long as GLDPC codes are generated as described above.



(a) (b)

Figure 2.3: In Fig. 2.3 (a), we plot the BER as a function of the channel erasure probability for a $(2, 6)$ base DD and a rate-1/2 Hamming $(6, 3)$ linear block code as component code. In Fig. 2.3 (b), we plot the BER as a function of the channel erasure probability for a $(2, 8)$ base DD and a rate-1/2 $(8, 4)$ Hamming component code. Results have been averaged over 10 generated samples from the $\mathcal{C}_{J,K,\nu}$ ensemble with a blocklength of $N = 10000$ bits.

2.3 Asymptotic analysis

The P-PD decoder yields a sequence of residual graphs by sequentially removing degree-one SPC nodes and decodable GC nodes from the GLDPC Tanner graph. Our next goal is to predict the asymptotic behaviour of the $\mathcal{C}_{J,K,\nu}$ ensemble under P-PD by extending the methodology proposed in [50] to analyze the asymptotic behavior of LDPC ensembles under PD. In [50], it is shown that if we apply the PD to elements of an LDPC ensemble, then the expected DD of the sequence of residual graphs can be described as the solution of a set of differential equations. Furthermore, the deviation of the process w.r.t. the expected evolution decreases exponentially fast with the LDPC blocklength. This analysis is based on a result on the evolution of Markov processes due to Wormald [103]. The proof that

the GLDPC asymptotic graph evolution under P-PD can be predicted using the same result is given in Appendix A.1. In this section, we introduce the notation used to characterize the DDs of the residual Tanner graphs of GLDPC ensembles with P-PD decoding and then present the system of differential equations that describes the asymptotic GLDPC graph evolution. In order to characterize the DDs of the residual Tanner graphs of GLDPC ensembles is to augment the DD notation introduced in Section 2.1 to differentiate between GC nodes that have been tagged as decodable and those tagged as non-decodable. In order to simplify the formulation, we restrict ourselves to the case $p_w = 0$ for $w \geq \mathtt{d}+2$, i.e., we consider component codes that can only decode a certain fraction of erasure patterns of degrees $\mathtt{d}$ and $\mathtt{d}+1$ and all erasure patterns of degree below $\mathtt{d}$. This may not be a strong assumption. After exhaustive search of short linear block component codes (blocklengths up to 15 bits), we have not found any component code with $p_w > 0$ for $w \geq \mathtt{d}+2$. In any case, the analysis provided here directly generalizes to any arbitrary p_w.

As introduced in Section 2.1, any edge adjacent to a degree i variable node is said to have left degree i, $i = 1, \ldots, J$. Similarly, any edge adjacent to a degree j SPC (GC) node is said to have right SPC (GC) degree j, $j = 1, \ldots, K$. Given the residual graph at the ℓ-th iteration of the P-PD algorithm, let $L_i^{(\ell)}$ denote the number of edges with left degree i at iteration ℓ. Similarly, let $R_{pj}^{(\ell)}$ denote the number of edges with right SPC degree j and $R_{cj}^{(\ell)}$ denote the number of edges with right GC degree j at iteration ℓ. For $j \in \{\mathtt{d}, \mathtt{d}+1\}$, we split $R_{cj}^{(\ell)}$ into two terms, $\hat{R}_{cj}^{(\ell)}$ and $\bar{R}_{cj}^{(\ell)}$, where $\hat{R}_{cj}^{(\ell)}$, $j \in \{\mathtt{d}, \mathtt{d}+1\}$ denotes the number of edges with right GC degree j connected to GC nodes tagged as *decodable*, and $\bar{R}_{cj}^{(\ell)}$ denotes the number of edges with right GC degree j connected to GC nodes tagged as *not-decodable*. Clearly, we have $R_{cj}^{(\ell)} = \hat{R}_{cj}^{(\ell)} + \bar{R}_{cj}^{(\ell)}, j = \mathtt{d}, \mathtt{d}+1$. Recall that $\mathtt{E}$ denotes the number of edges in the original GLPDC graph.

In the following theorem, we make use of Wormald's theorem [103] to show that the DD of the sequence of residual graphs during P-PD of a specific instance of the $\mathcal{C}_{J,K,\nu}$ ensemble converges to a function that can be computed by solving

a set of deterministic differential equations. More specifically, for any element $Z^{(\ell)} \in \{L_i^{(\ell)}, R_{pj}^{(\ell)}, R_{cj}^{(\ell)}\}_{\substack{i=1,\dots,J \\ j=1,\dots,K}}$ there exists a constant ξ such that

$$P\left(\left|Z^{(\ell)}/\mathtt{E} - z^{(\ell/\mathtt{E})}\right| > \xi \mathtt{E}^{-\frac{1}{6}}\right) = \mathcal{O}\left(\mathrm{e}^{-\sqrt{\mathtt{E}}}\right), \tag{2.16}$$

where $z^{(\ell/\mathtt{E})}$ is the solution of a set of differential equations for that element of the DD, and $\mathcal{O}\left(\mathrm{e}^{-\sqrt{\mathtt{E}}}\right)$ summarizes terms of order $\mathrm{e}^{-\sqrt{\mathtt{E}}}$. See Appendix A.1 for more details. In the following, we use the notation $Z^{(\ell)}/\mathtt{E} \to z^{(\ell/\mathtt{E})}$ to describe convergence in the sense of (2.16).

Theorem 4. Consider a BEC with erasure probability ϵ and assume we use elements of the $\mathcal{C}_{\overline{\lambda},\overline{\rho}_p,\overline{\rho}_c,\nu}$ code ensemble for transmission. If we use P-PD with parameters $(\mathtt{d}, p_{\mathtt{d}}, p_{\mathtt{d}+1})$, then the DD of the residual graph at iteration ℓ converges to

$$L_i^{(\ell)}/\mathtt{E} \to l_i^{(\tau)}, \quad i \in \{1,\dots,J\} \tag{2.17}$$

$$R_{pj}^{(\ell)}/\mathtt{E} \to r_{pj}^{(\tau)}, \quad j \in \{1,\dots,K\} \tag{2.18}$$

$$R_{cj}^{(\ell)}/\mathtt{E} \to r_{cj}^{(\tau)}, \quad j \in \{1,\dots,K\} \text{ and } j \notin \{\mathtt{d}, \mathtt{d}+1\} \tag{2.19}$$

$$\hat{R}_{cj}^{(\ell)}/\mathtt{E} \to \hat{r}_{cj}^{(\tau)}, \quad j \in \{\mathtt{d}, \mathtt{d}+1\} \tag{2.20}$$

$$\bar{R}_{cj}^{(\ell)}/\mathtt{E} \to \bar{r}_{cj}^{(\tau)}, \quad j \in \{\mathtt{d}, \mathtt{d}+1\} \tag{2.21}$$

where $l_i^{(\tau)}$, $r_{pj}^{(\tau)}$, $r_{cj}^{(\tau)}$, $\hat{r}_{cj}^{(\tau)}$, $\bar{r}_{cj}^{(\tau)}$, and $\tau = \frac{\ell}{\mathtt{E}} \in [0, \sum_{i=1}^{J} l_i^{(\tau)}/i]$ are the solutions to the following system of differential equations:

$$\frac{\mathrm{d}l_i^{(\tau)}}{\mathrm{d}\tau} = -\frac{i l_i^{(\tau)}}{e^{(\tau)}}\left(P_{p1}^{(\tau)} + \sum_{w=1}^{\mathtt{d}+1} w P_{cw}^{(\tau)}\right), \tag{2.22}$$

$$\begin{aligned}\frac{\mathrm{d}r_{pj}^{(\tau)}}{\mathrm{d}\tau} &= P_{p1}^{(\tau)}\left((r_{p(j+1)}^{(\tau)} - r_{pj}^{(\tau)})\frac{j(a^{(\tau)}-1)}{e^{(\tau)}} - \mathbb{I}[j=1]\right) \\ &\quad + \sum_{w=1}^{\mathtt{d}+1} P_{cw}^{(\tau)}(r_{p(j+1)}^{(\tau)} - r_{pj}^{(\tau)})\frac{jw(a^{(\tau)}-1)}{e^{(\tau)}},\end{aligned} \tag{2.23}$$

$$\frac{\mathrm{d}r_{cj}^{(\tau)}}{\mathrm{d}\tau} = P_{p1}^{(\tau)}\left((r_{c(j+1)}^{(\tau)} - r_{cj}^{(\tau)})\frac{j(a^{(\tau)}-1)}{e^{(\tau)}}\right)$$

$$+\sum_{w=1}^{\mathtt{d}+1} P_{cw}^{(\tau)} \left((r_{c(j+1)}^{(\tau)} - r_{cj}^{(\tau)}) \frac{jw(a^{(\tau)}-1)}{e^{(\tau)}} - w\mathbb{I}[j=w] \right), \quad j \notin \{\mathtt{d}, \mathtt{d}+1\} \tag{2.24}$$

$$\frac{\mathrm{d}\hat{r}_{cj}^{(\tau)}}{\mathrm{d}\tau} = P_{p1}^{(\tau)} \left((p_j \bar{r}_{c(j+1)}^{(\tau)} + \hat{r}_{c(j+1)}^{(\tau)} - \hat{r}_{cj}^{(\tau)}) \frac{j(a^{(\tau)}-1)}{e^{(\tau)}} \right)$$
$$+\sum_{w=1}^{j+1} P_{cw}^{(\tau)} \left((p_j \bar{r}_{c(j+1)}^{(\tau)} + \hat{r}_{c(j+1)}^{(\tau)} - \hat{r}_{cj}^{(\tau)}) \frac{jw(a^{(\tau)}-1)}{e^{(\tau)}} - w\mathbb{I}[w=j] \right), \quad j \in \{\mathtt{d}, \mathtt{d}+1\} \tag{2.25}$$

$$\frac{\mathrm{d}\bar{r}_{cj}^{(\tau)}}{\mathrm{d}\tau} = P_{p1}^{(\tau)} \left(((1-p_j) \bar{r}_{c(j+1)}^{(\tau)} - \bar{r}_{cj}^{(\tau)}) \frac{j(a^{(\tau)}-1)}{e^{(\tau)}} \right)$$
$$+\sum_{w=1}^{j+1} P_{cw}^{(\tau)} \left(((1-p_j) \bar{r}_{c(j+1)}^{(\tau)} - \bar{r}_{cj}^{(\tau)}) \frac{jw(a^{(\tau)}-1)}{e^{(\tau)}} - w\mathbb{I}[w=j] \right), \quad j \in \{\mathtt{d}, \mathtt{d}+1\} \tag{2.26}$$

In (2.22)-(2.26), $\mathbb{I}[\cdot]$ denotes the indicator function, and

$$e^{(\tau)} = \sum_{i=1}^{J} l_i^{(\tau)} = \sum_{j=1}^{K} [r_{pj}^{(\tau)} + r_{cj}^{(\tau)}], \tag{2.27}$$

$$a^{(\tau)} = \sum_{i} i l_i^{(\tau)} / e^{(\tau)}, \tag{2.28}$$

$$P_{p1}^{(\tau)} = \frac{r_{p1}^{(\tau)}}{s^{(\tau)}}, \tag{2.29}$$

$$P_{cj}(\tau) = \begin{cases} \dfrac{r_{cj}^{(\tau)}/j}{s^{(\tau)}} & j < \mathtt{d} \\ \dfrac{\hat{r}_{cj}^{(\tau)}/j}{s^{(\tau)}} & j \in \{\mathtt{d}, \mathtt{d}+1\} \end{cases} \tag{2.30}$$

$$s^{(\tau)} = r_{p1}^{(\tau)} + \sum_{w=1}^{\mathtt{d}-1} \frac{r_{cw}^{(\tau)}}{w} + \frac{\hat{r}_{c\mathtt{d}}^{(\tau)}}{\mathtt{d}} + \frac{\hat{r}_{c(\mathtt{d}+1)}^{(\tau)}}{\mathtt{d}+1}. \tag{2.31}$$

The initial conditions of the system of differential equations (2.22)-(2.26) are given by

$$l_i^{(0)} = \epsilon \lambda_i, \tag{2.32}$$

$$r_{pj}^{(0)} = \sum_{\alpha \geq j} \rho_{p\alpha} \binom{\alpha-1}{j-1} \epsilon^j (1-\epsilon)^{\alpha-j}, \tag{2.33}$$

$$r_{cj}^{(0)} = \sum_{\alpha \geq j} \rho_{c\alpha} \binom{\alpha - 1}{j - 1} \epsilon^j (1 - \epsilon)^{\alpha - j}, \tag{2.34}$$

$$\hat{r}_{c\nu}^{(0)} = p_\nu r_{c\nu}^{(0)}, \tag{2.35}$$

$$\bar{r}_{c\nu}^{(0)} = (1 - p_\nu) r_{c\nu}^{(0)} \tag{2.36}$$

for $i = 1, \ldots J$, $j = 1, \ldots, K$, and $\nu = \mathtt{d}, \mathtt{d}+1$.

Proof. See Appendix A.1. □

Using Theorem 4, we can predict the P-PD threshold for the $\mathcal{C}_{J,K,\nu}$ code ensemble by setting $\lambda_i = \mathbb{I}[i = J]$ in (2.32), $\rho_{p\alpha} = (1 - \nu)\mathbb{I}[\alpha = K]$ in (2.33), and $\rho_{c\alpha} = \nu \mathbb{I}[\alpha = K]$ in (2.34). We then numerically search for the highest ϵ value for which the function $r_{p1}^{(\tau)} + \sum_{w=1}^{\mathtt{d}-1} r_{cw}^{(\tau)}/w + \hat{r}_{c\mathtt{d}}^{(\tau)}/\mathtt{d} + \hat{r}_{c(\mathtt{d}+1)}^{(\tau)}/(\mathtt{d}+1)$ remains strictly positive for any $\tau \in [0, \sum_{i=1}^{J} l_i^{(\tau)}/i]$ such that $e^{(\tau)} > 0$.

2.3.1 An upper bound on the iterative-decoding threshold

For standard LDPC code ensembles, it is known that the BP iterative decoding threshold is upper bounded by the so-called *stability condition (STC)* [82]:

$$\epsilon^* \leq \left[\lambda_2 \ \rho'(1)\right]^{-1}, \tag{2.37}$$

where $\rho(x)$ is the right degree polynomial, $\rho'(1)$ its derivative at $x = 1$ and λ_2 is the fraction of edges in the graph with left degree equal to 2. In [78], Paolini, Fossorier, and Chiani extended the bound for GLDPC code ensembles by performing a Taylor expansion of the asymptotic GLDPC EXIT function. In particular, they proved that if the GLDPC code ensemble only contains generalized component codes with $\mathtt{d} \geq 3$, then the iterative decoding threshold is upper bounded by

$$\epsilon^* \leq \left[\lambda_2 \ \rho_p'(1)\right]^{-1}, \tag{2.38}$$

where

$$\rho_p(x) = \sum_{j \geq 2} \rho_{pj} x^{j-1}, \tag{2.39}$$

and ρ_{pj}, as defined in Section 2.1, is the fraction of edges in the GLDPC Tanner graph connected to degree-j SPC nodes. For the $\mathcal{C}_{J,K,\nu}$ ensemble with $J = 2$, this bound simplifies to

$$\epsilon^* \leq \frac{1}{(K-1)(1-\nu)}, \tag{2.40}$$

while for $J > 2$ this bound is non-informative (it is infinite) since $\lambda_2 = 0$.

2.4 Analysis of the $\mathcal{C}_{J,K,\nu}$ ensemble under P-PD

In this section, we study the asymptotic performance of the $\mathcal{C}_{J,K,\nu}$ ensemble for different base DDs as we vary the fraction ν of GC nodes in the graph. We use high rate base DDs that correspond to regular LDPC code ensembles with variable degree equal to $J = 2$. Further examples with $J > 2$ are discussed in Sections 2.5.2 and 2.7. We summarize the parameter of the base DD considered here in Table 3.1. We denote by ϵ_0 the PD threshold of the base LDPC ensemble. Recall that $p_w = 1$ for $w \leq \mathtt{d} - 1$ and $p_w = 0$ for $w \geq \mathtt{d} + 2$. In order to determine $p_{\mathtt{d}}, p_{\mathtt{d}+1}$, we performed an exhaustive search over the database [29, 30], which implements MAGMA [8] to design block codes with the largest minimum distance. For every K, we search for the code with the largest minimum distance $\mathtt{d}$, and we use the corresponding $p_{\mathtt{d}}$ and $p_{\mathtt{d}+1}$ parameters. Like this, we ensure that there exists at least one linear block code that satisfies these requirements. We use this specific block code as the reference of a family of linear block codes with the same decoding capabilities. The values found are listed in Table 3.2 and used as a reference for a whole family of linear block codes. The corresponding reference block codes are listed in Appendix C.1. Note that despite having different blocklength and rate, many reference block codes share the same $p_{\mathtt{d}}$, $p_{\mathtt{d}+1}$ parameters.

We construct $\mathcal{C}_{J,K,\nu}$ ensembles by combining various base DDs with the component code families summarized in Table 3.2. For each code ensemble, we compute the P-PD threshold ϵ^* as a function of ν.

Table 2.1: Base DDs, their design rates and iterative decoding thresholds under PD

Base DD	K	$\mathtt{R}_0$	ϵ_0	Gap to capacity $(1-\mathtt{R}_0-\epsilon_0)$
$(2,6)$-regular	6	2/3	0.206	0.127
$(2,7)$-regular	7	5/7	0.167	0.119
$(2,8)$-regular	8	3/4	0.147	0.103
$(2,15)$-regular	15	13/15	0.071	0.062

Table 2.2: Families of component linear block codes.

Code Family Index	blocklength K	$\mathtt{d}$	$p_{\mathtt{d}}$	$p_{\mathtt{d}+1}$
I	6	3	0.8	0
II	6	4	0.8	0
III	7	3	0.8	0
IV	7	4	0.8	0
V	8	4	0.8	0
VI	8	4	0.9143	0.5714
VII	8	5	0.9643	0.75
VIII	15	3	0.9231	0.6154
IX	15	4	0.9231	0.6154

2.4.1 Results for $(2,6)$ and $(2,7)$ base DDs

Fig. 2.4 shows the computed P-PD threshold ϵ^* of the $\mathcal{C}_{J,K,\nu}$ ensemble for a base DD $(2,6)$-regular as a function of ν. We consider GC nodes with minimum distance $\mathtt{d}$ equal to 3 and 4 and parameters given by Families I and II in Table 3.2. We also include the BD-PD threshold, which only depends on the minimum distance $\mathtt{d}$ of the component codes and can be computed by solving the system of differential equations in Theorem 4 by setting $p_{\mathtt{d}} = p_{\mathtt{d}+1} = 0$. First of all, observe that the P-PD gains in threshold w.r.t. BD-PD are only significant for large values of ν. Furthermore, for both P-PD and BD-PD, using component codes with larger minimum distance ($\mathtt{d} = 4$ instead of $\mathtt{d} = 3$) pays off only for very large values of ν.

Since increasing ν also modifies the code rate $\mathtt{R}(\nu)$ in (2.4), the comparison

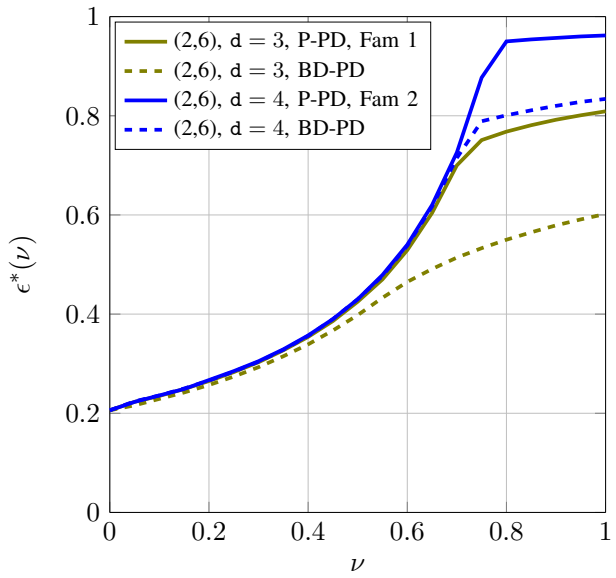
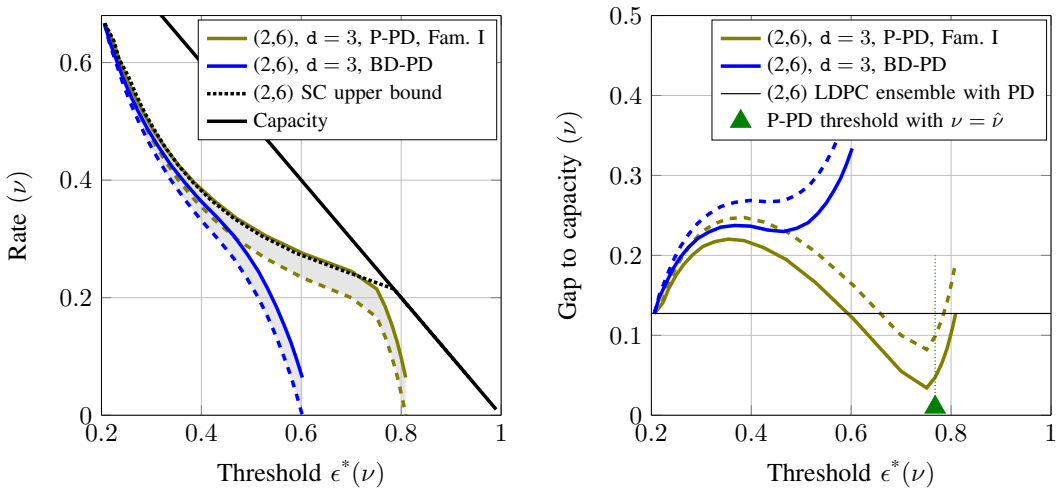


Figure 2.4: P-PD and BD-PD thresholds as a function of ν for the $(2, 6)$ base DD.

in Fig. 2.4 can be misleading, as we cannot directly evaluate the distance to the channel capacity. In fact, not all values of ν are achievable, since they would give rise to a negative rate $\mathtt{R}(\nu)$. We overcome this issue by directly comparing the asymptotic threshold and code rate, both defined as parametric curves w.r.t. ν. Denote by $\epsilon^*(\nu)$ the threshold ϵ^* as a function of $\nu \in [0, 1]$. From Fig. 2.4 we see that $\epsilon^*(\nu)$ is a continuous, strictly increasing function of ν and that for $\nu = 0$ its value is equal to ϵ_0, the threshold of the base LDPC ensemble. The inverse of this function, which can be obtained numerically, is denoted by $\nu(\epsilon^*)$ and provides the minimum fraction of GC nodes in the graph required to achieve an ensemble threshold at least ϵ^*. Given the function $\nu(\epsilon^*)$ described above, we use Lemma 2 to determine bounds on $\mathtt{R}(\nu)$ for a given targeted decoding threshold ϵ^*. More precisely, by using $\nu(\epsilon^*)$ in (2.6), we obtain a *converse bound* on the coding rate required to achieve a P-PD decoding threshold equal to ϵ^* using component codes with minimum distance $\mathtt{d}$. Similarly, using $\nu(\epsilon^*)$ in (2.7), we obtain an *achievable bound* on the coding rate required to achieve a P-PD decoding threshold equal to ϵ^* using linear component codes with minimum distance $\mathtt{d}$. We proceed along the same lines to obtain bounds on the $\mathcal{C}_{J,K,\nu}$ rate for the BD-PD thresholds.



(a) (b)

Figure 2.5: In Fig. 2.5 (a), we plot the bounds on the $\mathcal{C}_{J,K,\nu}$ coding rate in (2.6) and (2.7) for the base DD $(2, 6)$ and component codes of minimum distance $\mathtt{d} = 3$ as a function of the P-PD and BD-PD thresholds. In Fig. 2.5 (b), we show their gap to channel capacity. We also indicate the P-PD threshold for $\nu = \hat{\nu}$.

In Fig. 2.5 (a) we plot these bounds as a function of ϵ^*, both for P-PD and BD-PD, using Code Family I component codes with minimum distance $\mathtt{d} = 3$. We further include the STC upper bound in (2.40). Observe that (2.40) coincides with the rate-threshold converse bound in (2.6) up to $\nu \approx 0.75$. Above $\nu = 0.8$, the STC bound exceeds channel capacity.

In Fig. 2.5 (b), we show the gap to channel capacity computed for each case, and indicate the threshold $\epsilon^*(\hat{\nu})$ with $\hat{\nu}$ given in (2.15). Since $\epsilon^*(\nu)$ is monotonically increasing in ν, any configuration with threshold larger than $\epsilon^*(\hat{\nu})$ has a minimum distance that grows linearly with the block length N. Observe that the performance of both BD-PD and P-PD overlaps for coding rates close to the original rate of the base DD, i.e., for small values of ν. However, as $\epsilon^*(\nu)$ increases, P-PD significantly outperforms BD-PD. Furthermore, there are values of ν for which the gap to capacity of P-PD is smaller than that for the base LDPC ensemble under PD. For the $(2, 6)$ base DD, the minimum gap to capacity of P-PD, measured using the achievable rate bound, is 0.0823 for a coding rate of 0.1667.

For $\nu = \hat{\nu}$, the gap to capacity grows to 0.0987 but it is still below the base LDPC gap to capacity, which is 0.1273 according to Table 3.1. Thus, for ν slightly above $\hat{\nu}$ we are able to reduce the original gap to capacity and at the same time obtain a good ensemble with respect to minimum distance. Observe also that the region where the $\mathcal{C}_{J,K,\nu}$ ensemble outperforms the base LDPC ensemble is very narrow, and it does not include the case where all check nodes are GC nodes ($\nu = 1$).

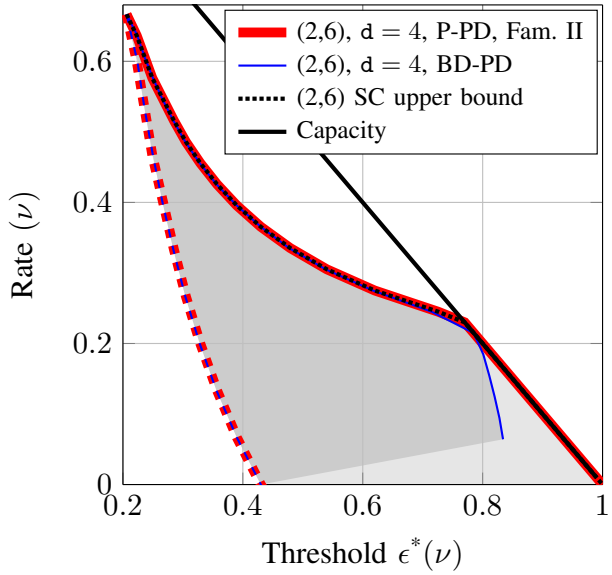
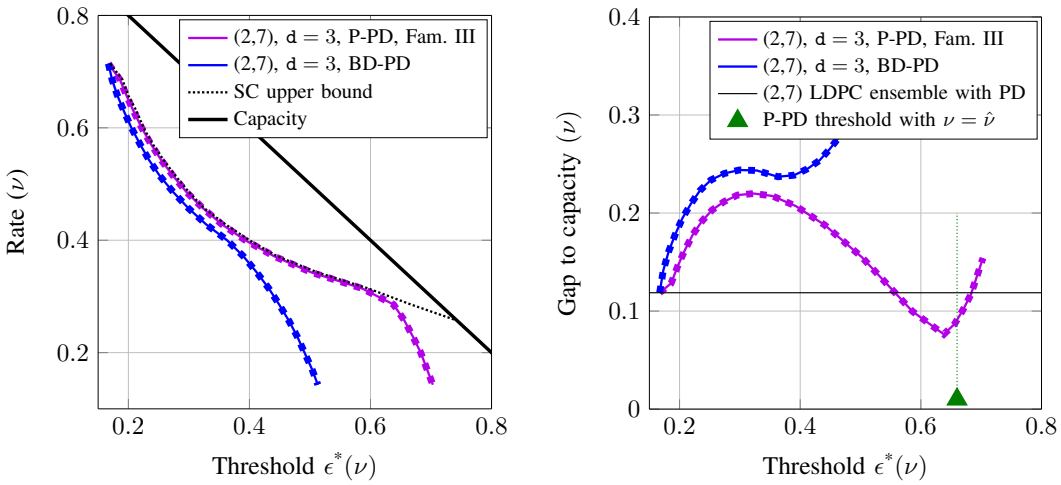


Figure 2.6: Bounds on the $\mathcal{C}_{J,K,\nu}$ coding rate in (2.6) and (2.7) for a base DD $(2, 6)$ and $\mathtt{d} = 4$ component codes as a function of the P-PD and BD-PD thresholds.

Fig. 2.6 reproduces the results for the Code Family II with minimum distance $\mathtt{d} = 4$. However, in this case the two bounds are loose and it is uncertain whether we can find a specific component code in the family that is able to operate close to the converse bound. The P-PD converse bound now overlaps with the STC bound in the whole regime and, for large $\epsilon^*(\nu)$, it coincides with the capacity. Furthermore, the bounds for P-PD and BD-PD overlap in a large region despite the fact that P-PD using component codes from Family II resolves degree-$\mathtt{d}$ erasure patterns with high probability (0.8).

In Fig. 2.7 we show the asymptotic behaviour of the $\mathcal{C}_{J,K,\nu}$ ensemble constructed using a $(2, 7)$ base DD with $\mathtt{d} = 3$ component codes. As predicted by

Lemma 2, when using component codes of blocklength $K = 7$ with minimum distance $\mathtt{d} = 3$, the converse and achievable bound on the $\mathcal{C}_{J,K,\nu}$ coding rate coincide. Thus, the existence of a linear block component code that satisfies the properties of Code Family III and for which the $\mathcal{C}_{J,K,\nu}$ ensemble asymptotically achieves the results in Fig. 2.7 is guaranteed. Again, there is a region where the gap to capacity of P-PD can be reduced with respect to that of the base LDPC ensemble, which is roughly aligned with the point where the P-PD threshold separates from the STC upper bound in (2.40).



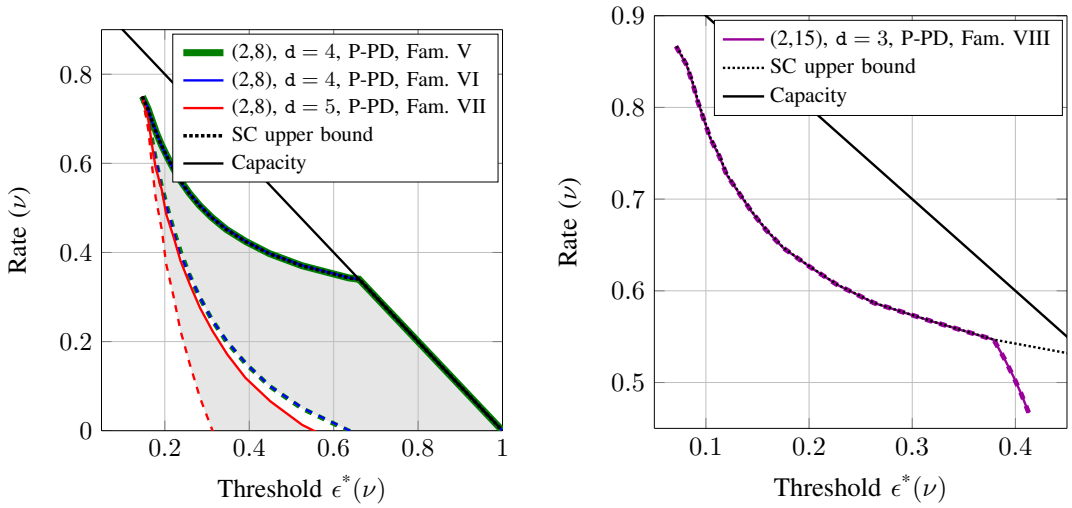
(a) (b)

Figure 2.7: In (a), we plot the bounds on the $\mathcal{C}_{J,K,\nu}$ coding rate in (2.6) and (2.7) for a base DD $(2, 7)$ as a function of the P-PD and BD-PD thresholds. Note that the bounds overlap in this case. In (b), we show the gap to channel capacity for each case. We also indicate the P-PD threshold for $\nu = \hat{\nu}$.

2.4.2 Results for higher-density base DDs

We finish this section by extending the above results to base DDs with higher check degree and, thus higher ensemble density. In Fig. 2.8(a), we show the asymptotic behavior of the $\mathcal{C}_{J,K,\nu}$ ensemble constructed using a $(2, 8)$ base DD with component codes in Code Families V, VI and VII (See Table 3.2). Observe first that the rate bounds for Code Families V and VI coincide, even though Code Family VI has

better decoding capabilities. In both cases the bounds are loose, but we can still observe a significant improvement w.r.t. the Code Family VII, which has very large ($\mathtt{d} = 5$) minimum distance and, hence, and small coding rate. This again illustrates the trade-off between the threshold performance and the rate penalty induced by considering lower rate GC nodes. In Fig. 2.8(b), we consider a $(2, 15)$ base DDs with a component code of Code Family VIII ($\mathtt{d} = 3$). In this case, as predicted by lemma 2, the bounds coincide and the gap to capacity is minimized at a coding rate $\mathtt{R} \approx 0.54$ and threshold $\epsilon^* \approx 0.379$, resulting in a gap capacity equal to 0.074. This is slightly above the gap to capacity for the base LDPC ensemble (0.062). Also, at this point the GLDPC ensemble does not have linear growth of the minimum distance, since for this ensemble, $\epsilon^*(\hat{\nu}) = 0.493$.



(a) (b)

Figure 2.8: We plot the bounds on the $\mathcal{C}_{J,K,\nu}$ coding rate in (2.6) and (2.7) for a base DD $(2, 8)$ (Fig. 2.8 (a)) and $(2, 15)$ (Fig. 2.8 (b)) as a function of the P-PD threshold.

2.5 Selecting specific component codes

By using the bounds on the $\mathcal{C}_{J,K,\nu}$ code rate, we have been able to assess the performance of $\mathcal{C}_{J,K,\nu}$ ensembles for a family of linear component codes. In certain scenarios the proposed bounds on the $\mathcal{C}_{J,K,\nu}$ code rate provide meaningful design

information about the asymptotic behavior of the ensemble. The natural question that arises at this point is whether we can find specific component codes within the family that outperform the achievable bound in (2.7), reducing the gap to the rate converse bound in (2.6). In this section, we analyze the asymptotic performance of $\mathcal{C}_{J,K,\nu}$ when component codes are chosen from the the list of reference linear block component codes summarized in Table 2.3. The construction of these linear block codes is detailed in [29], and their generator matrix is given in Appendix C.1. We use the notation R-I to denote the reference linear block code of Code Family I.

Table 2.3: Reference component codes. The parameter `k` describes the number of rows in the parity-check matrix.

Code index	Blocklength K	`k`	Rate	Code family in Table 3.2
R-I	6	3	1/2	I
R-II	6	4	1/3	II
R-III	7	3	4/7	III
R-IV	7	4	3/7	IV
R-V	8	4	1/2	V
R-VI	8	5	3/8	VI
R-VII	8	6	1/4	VII
R-VIII	15	4	11/15	VIII
R-IX	15	5	2/3	IX

Once we fix a particular class of component codes to be used at GC nodes, we can replace the $\mathcal{C}_{J,K,\nu}$ code bounds by the actual code rate in (2.4). In Fig. 2.9 we plot the $\mathcal{C}_{J,K,\nu}$ coding rate (using markers), and the STC upper bound and and the achievable bound of the corresponding family of codes for $(2, 6)$ and $(2, 7)$ base DDs. Results for $(2, 8)$ and $(2, 15)$ base DDs can be found in Fig. 2.10. Observe that, with the proposed component codes, we are able to perform at least as good as the achievable bound of the corresponding family of block component codes. In some cases, e.g. the $(2, 8)$ base DD, the achievable bound is significantly outperformed. Recall that for the $(2, 8)$ base DD the rate bounds in Fig. 2.8(a)

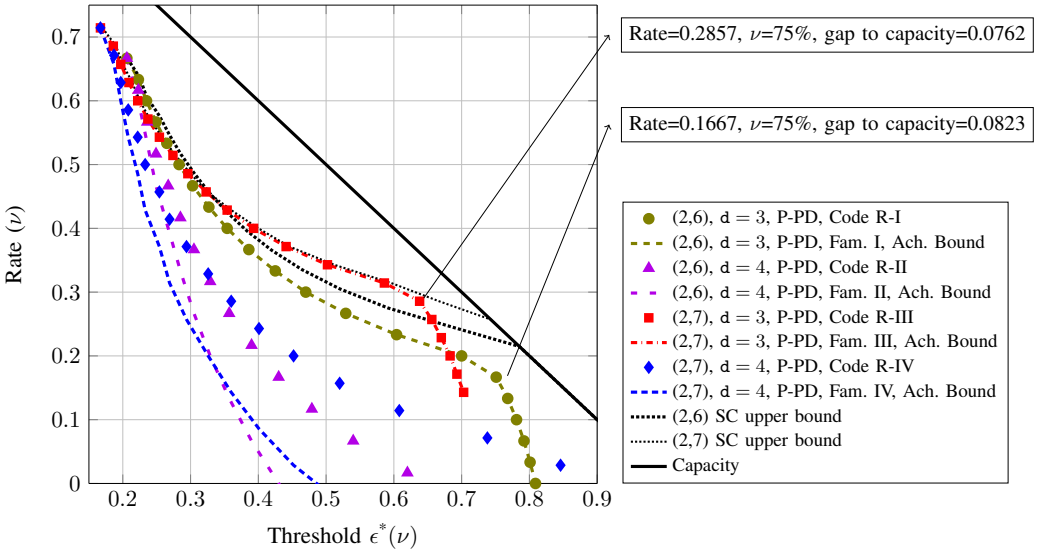


Figure 2.9: $\mathcal{C}_{J,K,\nu}$ coding rate and achievable bound in (2.7) for $(2,6)$ and $(2,7)$ base DDs and component codes from Table 3.2 and 2.3 as a function of the P-PD decoding threshold.

are loose. While for the $(2,7)$ and $(2,15)$ codes the STC bound is attained except for large values of ν, for the $(2,6)$ and $(2,8)$ ensembles results suggest that there is still room for improving the component code design.

Finally, in the same figures, we highlight those points for which, asymptotically, the $\mathcal{C}_{J,K,\nu}$ ensemble with the proposed linear component codes under P-PD operates closer to channel capacity than the base LDPC code ensemble under PD. For both the $(2,6)$, $(2,7)$, and the $(2,8)$ base DDs we were able to find such points. For the $(2,15)$ ensemble, the minimum gap to capacity obtained is slightly above the one of the base LDPC code ensemble under PD (0.0743 and 0.0623 respectively).

2.5.1 Growth Rate of the Weight Distribution

Upon selecting a specific block code, we can compute the weight spectral shape $G(\alpha)$ in (2.14) using the tools proposed in [24]. In Fig. 2.11, we plot $G(\alpha)$ for different values of ν, computed for the $(2,6)$ base DD with Code R-I as component code (Fig. 2.11 (a)) and the $(2,7)$-regular base DD with Code R-III as component code (Fig. 2.11 (b)). Recall that the critical exponent codeword weight ratio is defined as $\hat{\alpha} \triangleq \inf\{\alpha \geq 0 | G(\alpha) \geq 0\}$. In the plots, we highlight $\hat{\alpha}$ with a star. By

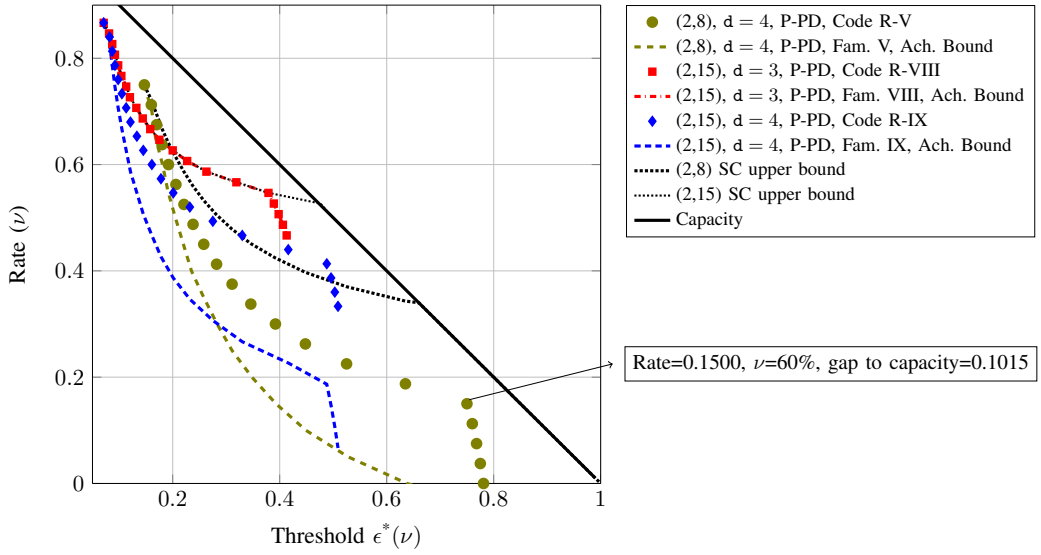
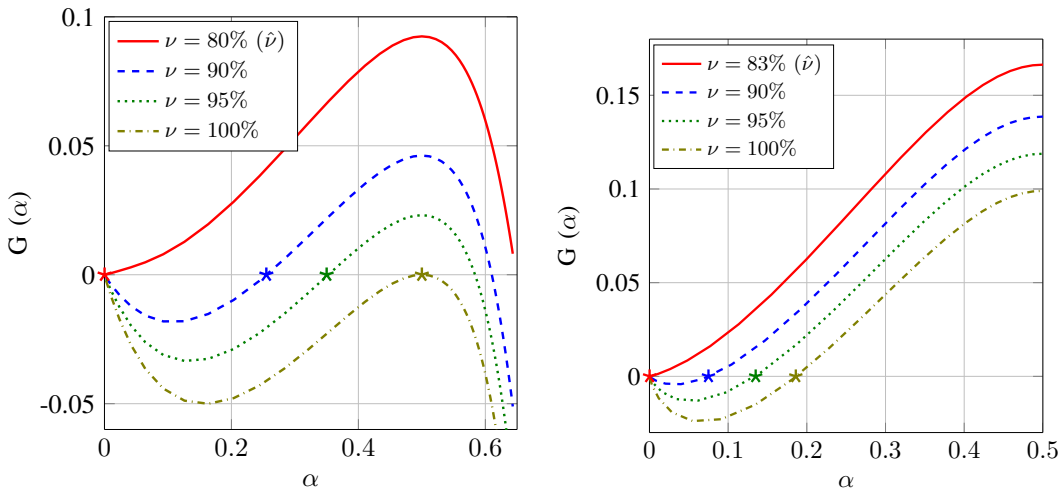


Figure 2.10: $\mathcal{C}_{J,K,\nu}$ coding rate and achievable bound in (2.7) for $(2,8)$ and $(2,15)$ base DDs and component codes from Tables 3.2 and 2.3 as a function of the P-PD decoding threshold.

Lemma 3, we have $\hat{\alpha} = 0$ at $\nu = \hat{\nu}$. As ν grows, $\hat{\alpha}$ grows, too, and it achieves its maximum at $\nu = 1$. These results indicate that there is a trade-off between the gap to capacity and $\hat{\alpha}(\nu)$, the critical exponent codeword weight ratio. As an example, we include values of both quantities in Table 2.4 for the $(2,6)$-regular base DD with Code R-I as component code.

Table 2.4: $\hat{\alpha}$, ϵ^* and Gap to capacity for different values of ν, computed for the $(2,6)$-base DD with Code R-I component codes

ν	α	ϵ^*	Gap to capacity
80%	0	0.768	0.0987
87.5%	0.2049	0.788	0.1287
90%	0.2556	0.792	0.1413
92.5%	0.3038	0.797	0.1530
95%	0.3526	0.801	0.1657
97.5%	0.4056	0.806	0.1773
100%	0.6078	0.809	0.1910



(a) (b)

Figure 2.11: In Fig. 2.11 (a), we plot the weight spectral shape $G(\alpha)$ in (2.14) of the $\mathcal{C}_{J,K,\nu}$ ensemble for a $(2, 6)$ base DD and with Code R-I as component code. In Fig. 2.11 (b), we plot the same quantity for the $\mathcal{C}_{J,K,\nu}$ ensemble for a $(2, 7)$ base DD and with Code R-II as component code (b).

2.5.2 Extension to irregular GDLPC code ensembles

To finish this section, we present some further examples using GLDPC code ensembles with irregular DD. Note that the initial conditions in (2.32)-(2.36) of the P-PD asymptotic analysis presented in Section 2.3 already consider an arbitrarily irregular DD, and hence the methodology presented is directly applicable to irregular GLDPC code ensembles. As an example, here we discuss two irregular GLDPC code ensembles:

- *Ensemble I* [77]. Rate 1/3, $\lambda(x) = 0.2x + 0.7118x^2 + 0.0882x^4$, $\nu^* = 0.6719$ and Hamming $(7, 4)$ component codes. Using ML decoding at GC nodes, the reported threshold is 0.540.

- *Ensemble II* [31]. Rate 1/2, $\lambda(x) = 0.80x^2 + 0.01x^5 + 0.01x^7 + 0.18x^9$, $\nu^* = 0.40$ and Hamming $(15, 11)$ component codes. Using ML decoding at GC nodes, the reported threshold is 0.466.

These ensembles have been constructed using numerical-constrained optimization

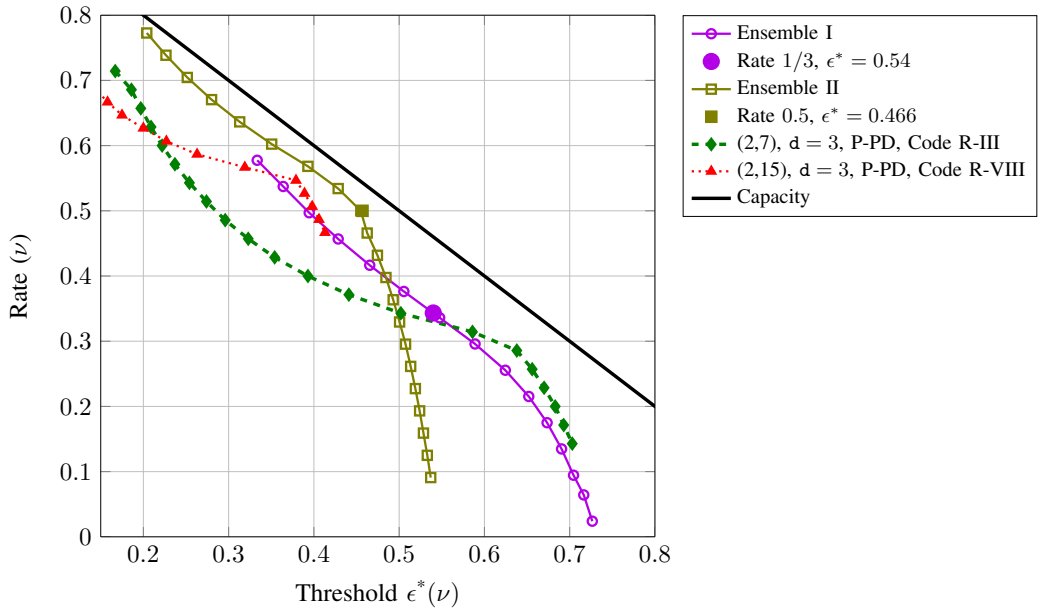


Figure 2.12: P-PD asymptotic threshold and coding rates for different regular and irregular GLDPC code ensembles with varying fraction ν of GC nodes in the graph.

methods. In Fig. 2.12 we show the results of the P-PD asymptotic analysis when we vary ν around the fraction ν^* defined above for each case. Observe first that in both cases our results are consistent with the thresholds computed in [77, 31]. In addition, they show that the gap to capacity for Ensemble II can be reduced if we slightly reduce the ensemble rate, i.e. by reducing ν to roughly 35% instead of 40%. For Ensemble I, the gap to capacity is indeed minimized at exactly the point predicted in [77]. For comparison, we have included $(2, X)$-regular GLDPC code ensembles with the same check node degrees (and thus same graph density) as Ensembles I and II. Observe that while Ensemble II significantly outperforms the rate-threshold tradeoff of the $(2, 15)$-GLDPC code ensemble with Code R-VIII as component code, the $(2, 7)$-regular GLDPC code with Code R-III as component code approximately attains threshold 0.540 at rate $\mathtt{R} = 1/3$, but can reduce the gap to capacity as we decrease the coding rate.

2.6 Random puncturing

We have proposed the P-PD algorithm as a flexible model to analyze beyond-BD decoding algorithm at GC nodes. Observe that for the P-PD algorithm, the evaluation of the coding rate and the iterative decoding threshold are decoupled problems. This provides a flexible analysis framework that allows the exploration of additional techniques to modify the designs presented above and further reduce the gap to capacity. In this section and the following one, we consider two relevant examples. Specifically, in this section we consider the use of random puncturing to accommodate the coding rate by dropping the transmission of a fraction of coded bits [66]. In the next section, a simple model of doubly-generalized LDPC (DG-LDPC) code ensembles is analyzed [101, 102, 22].

As illustrated in [66], a linear code is *punctured* by removing a set of columns from its generator matrix. After puncturing at random a fraction ξ of the coded bits in the $\mathcal{C}_{J,K,\nu}$ ensemble, the resulting coding rate is

$$\mathtt{R}(\nu,\xi) = \frac{\mathtt{R}(\nu)}{1-\xi}, \quad \xi \in [0,1), \tag{2.41}$$

where we recall that $\mathtt{R}(\nu)$ denotes the coding rate of the original $\mathcal{C}_{J,K,\nu}$ ensemble. In [66], the authors derive a simple analytic expression for the iterative belief propagation (BP) decoding threshold of a randomly punctured LDPC code ensemble on the binary erasure channel (BEC). Following their proof, it can be verified that the same results apply to a randomly punctured GLDPC code ensemble. The result reads as follows. Given a $\mathcal{C}_{J,K,\nu}$ ensemble with iterative decoding threshold $\epsilon^*(\nu)$, the threshold $\epsilon^*(\nu,\xi)$ of the GLDPC ensemble that follows by randomly puncturing a fraction ξ of the coded bits is related to the unpunctured case as follows:

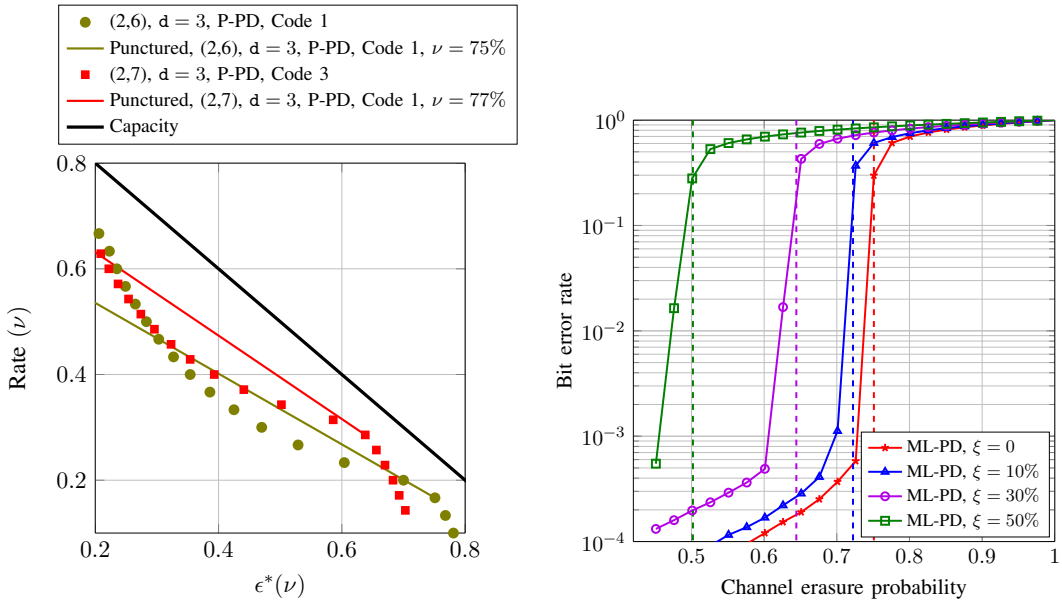
$$\epsilon^*(\nu,\xi) = 1 - \frac{1-\epsilon^*(\nu)}{1-\xi}. \tag{2.42}$$

Observe that the larger the unpunctured threshold $\epsilon^*(\nu)$ is, the larger the threshold of the punctured ensemble will be. In this regard, we can think of the design of a punctured GLDPC ensemble as a two stage process: First, the GLDPC code

ensemble can be designed by choosing ν to minimize the gap to capacity. Second, for a fixed ν, we can analyze the overall gap to capacity as we increasing the code rate by combining (2.41) and (2.42). We perform this experiment in Fig. 2.13 (a) for the $(2, 6)$ and the $(2, 7)$ base DDs and component codes R-I and R-III, respectively. With markers we show the $\mathcal{C}_{J,K,\nu}$ threshold-rate curve as we increase the fraction of GC nodes in the graph. Solid lines indicate the evolution of the rate and threshold of the punctured ensemble for fixed ν as we increase the puncturing fraction ξ. Observe that with puncturing it is possible to increase the coding rate and obtain an iterative decoding threshold that is closer to capacity than those obtained by the original $\mathcal{C}_{J,K,\nu}$ ensemble. The accuracy of the predicted threshold can be observed in Fig. 2.13 (b), where we include both the threshold predicted by (2.42) (dashed lines) and the simulated P-PD performance for the $(2, 6)$ base DD with component code R-I, $N = 10000$ bits, and different values of the puncturing rate ξ (solid lines). We note that, once we introduce puncturing, the STC upper bound in (2.40) is not applicable anymore.

2.7 Doubly-generalized LDPC codes

A different technique that can potentially help to find a better balance between coding rate and threshold is the inclusion of generalized variable nodes, giving rise to a doubly-generalized LDPC code ensemble [101]. In this section we develop an example with a simple class of a DG-LDPC ensemble. We modify the $\mathcal{C}_{J,K,\nu}$ ensemble by replacing a certain fraction β of regular variable (RV) nodes by generalized variable (GV) nodes, see Fig. 2.14. Degree-J RV nodes in the $\mathcal{C}_{J,K,\nu}$ graph can be seen as rate $1/J$ repetition code of block length J, where the input to the repetition code represents one bit of the DG-LDPC codeword. On the other hand, degree-J GV nodes are characterized by a $(J, \mathtt{m})$ linear block code, where the input to the variable component code represents $\mathtt{m}$ bits of the DG-LDPC codeword. Thus, the total block length of the DG-LDPC code ensemble is $N' = (1 - \beta)N + \beta N\mathtt{m}$, where N is the number of variable nodes (both RV and GV) in the graph. In the following, we will assume $J = 3, \mathtt{m} = 2$ and the following



(a) (b)

Figure 2.13: In Fig. 2.13 (a), we show with markers the $\mathcal{C}_{J,K,\nu}$ threshold-rate curve for the $(2,6)$ and the $(2,7)$ base DDs and component codes R-1 and R-3, respectively. Solid lines indicate the evolution of the rates and thresholds of the punctured ensemble for a fixed ν as we increase the puncturing fraction ξ. In Fig. 2.13 (b), we show the simulated P-PD performance for the $(2,6)$ base DD with component codes R-1, $N = 10000$ bits, and different values of the puncturing rate ξ. Vertical dashed lines indicate the thresholds predicted by (2.42).

generator matrix for GV nodes:

$$\mathrm{G} = \begin{pmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \end{pmatrix}. \tag{2.43}$$

Thus, each GV node encodes two bits of the DG-LDPC codeword. Denote this ensemble by $\mathcal{C}_{3,K,\nu,\beta}$. If the component codes at GC nodes are linear block codes with a $\mathtt{k}$-row parity check matrix, an easy calculation shows that the coding rate of the ensemble is

$$\mathtt{R}(\alpha,\beta) = 1 - (1-\mathtt{R}_0)\left(\frac{1+(\mathtt{k}-1)\nu}{1+\beta}\right). \tag{2.44}$$

As before, we characterize the component codes at GC nodes by the triple

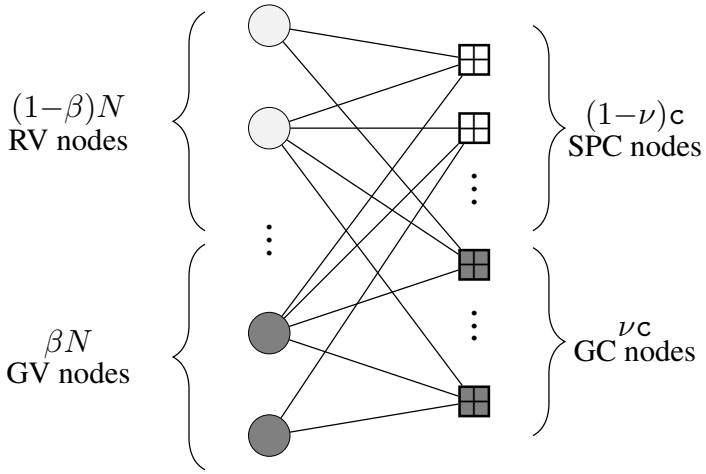


Figure 2.14: Tanner graph of the DG-LDPC code ensemble.

$(\mathtt{d}, p_{\mathtt{d}}, p_{\mathtt{d}+1})$. Furthermore, the code associated with the generator matrix (2.43) has minimum distance 2 and can only decode erasure patterns of weight one.

2.7.1 Decoding via P-PD

Suppose we use a random sample of the $\mathcal{C}_{3,K,\nu,\beta}$ code ensemble to transmit over a BEC(ϵ). RV nodes are removed from the graph with probability $1-\epsilon$. Regarding GV nodes, we have to consider the following three scenarios:

- With probability $(1-\epsilon)^2$ the two DG-LDPC coded bits are correctly received and the GV node can be removed from the graph.
- With probability $2\epsilon(1-\epsilon)$, only one of the two coded bits is received. Since the node is only encoding one unknown bit, note that we can replace the GV node in the graph by a degree-2 RV node.
- With probability ϵ^2 the GV node remains in the graph as a degree-3 GV node.

Decoding will be performed via P-PD. Since the code spanned by (2.43) can only decode one error, during the P-PD procedure every GV node needs to lose at least two edges before it can be removed from the graph. Further, once it loses one

edge, it can be replaced by a degree-2 RV node. Hence, a small modification is required at step 2) in the P-PD Algorithm in Section 2.2. Now, it reads as follows:

> 2) Remove from the Tanner graph the check node with the index drawn in Step 1). Further remove all connected RV nodes, connected degree-2 GV nodes and all attached edges.

2.7.2 Degree Distribution and Asymptotic Analysis

While no change is needed to describe the evolution of the check nodes of the residual DG-LDPC code ensemble during P-PD, additional definitions at the variable side are needed to tackle both RV nodes and GV nodes. Let $L_{r2}^{(\ell)}$ and $L_{r3}^{(\ell)}$ represent the total number of edges in the graph connected to RV nodes of degree 2 and 3, respectively, after iteration ℓ of the decoder. Further let $L_{g3}^{(\ell)}$ be the total number of edges in the graph connected to GV nodes of degree 3.

Theorem 5. *Consider a BEC with erasure probability ϵ and assume we use elements of the $\mathcal{C}_{3,K,\nu,\beta}$ code ensemble for transmission. If we use P-PD with parameters $(\mathtt{d}, p_{\mathtt{d}}, p_{\mathtt{d}+1})$, then the DD of the residual graph at iteration ℓ converges in the sense of* (2.16) *to*

$$L_{r2}^{(\ell)}/\mathtt{E} \to l_{r2}^{(\tau)}, \tag{2.45}$$

$$L_{r3}^{(\ell)}/\mathtt{E} \to l_{r3}^{(\tau)}, \tag{2.46}$$

$$L_{g3}^{(\ell)}/\mathtt{E} \to l_{g3}^{(\tau)}, \tag{2.47}$$

$$R_{pj}^{(\ell)}/\mathtt{E} \to r_{pj}^{(\tau)}, \quad j \in \{1, \ldots, K\} \tag{2.48}$$

$$R_{cj}^{(\ell)}/\mathtt{E} \to r_{cj}^{(\tau)}, \quad j \in \{1, \ldots, K\} \text{ and } j \notin \{\mathtt{d}, \mathtt{d}+1\} \tag{2.49}$$

$$\hat{R}_{cj}^{(\ell)}/\mathtt{E} \to \hat{r}_{cj}^{(\tau)}, \quad j \in \{\mathtt{d}, \mathtt{d}+1\} \tag{2.50}$$

$$\bar{R}_{cj}^{(\ell)}/\mathtt{E} \to \bar{r}_{cj}^{(\tau)}, \quad j \in \{\mathtt{d}, \mathtt{d}+1\} \tag{2.51}$$

where $l_{r2}^{(\tau)}$, $l_{r3}^{(\tau)}$, $l_{g3}^{(\tau)}$ $r_{pj}^{(\tau)}$, $r_{cj}^{(\tau)}$, $\hat{r}_{cj}^{(\tau)}$, $\bar{r}_{cj}^{(\tau)}$, $\tau = \frac{\ell}{\mathtt{E}} \in [0, \sum_{i=1}^{J} l_i^{(\tau)}/i]$ are the solutions to the system of differential equations given by (2.22)-(2.26) *using $a^{(\tau)} = (3l_{r3}^{(\tau)} +$*

$2l_{r2}^{(\tau)} + l_{g3}^{(\tau)})/e^{(\tau)}$ *and*

$$\frac{dl_{r2}^{(\tau)}}{d\tau} = 2\left(\frac{l_{g3}^{(\tau)} - l_{r2}^{(\tau)}}{e^{(\tau)}}\right)\left(P_{p1}^{(\tau)} + \sum_{w=1}^{\mathtt{d}+1} wP_{cw}^{(\tau)}\right) \tag{2.52}$$

$$\frac{dl_{r3}^{(\tau)}}{d\tau} = -\frac{3l_{r3}^{(\tau)}}{e^{(\tau)}}\left(P_{p1}^{(\tau)} + \sum_{w=1}^{\mathtt{d}+1} wP_{cw}^{(\tau)}\right) \tag{2.53}$$

$$\frac{dl_{g3}^{(\tau)}}{d\tau} = -\frac{3l_{g3}^{(\tau)}}{e^{(\tau)}}\left(P_{p1}^{(\tau)} + \sum_{w=1}^{\mathtt{d}+1} wP_{cw}^{(\tau)}\right), \tag{2.54}$$

Here, $e^{(\tau)}$*,* $P_{p1}^{(\tau)}$ *and* $P_{cw}^{(\tau)}$ *are defined in* (2.27)*,* (2.29)*, and* (2.30)*, respectively. The initial conditions of the system of differential equations in* (2.22)-(2.26) *and* (B.7)-(B.9) *are given by*

$$l_{g3}^{(0)} = \epsilon^2\beta, \tag{2.55}$$

$$l_{r3}^{(0)} = \epsilon(1-\beta), \tag{2.56}$$

$$l_{r2}^{(0)} = 4\beta\epsilon(1-\epsilon)/3 \tag{2.57}$$

and by (2.33)-(2.36) *evaluated at* $\epsilon' = \epsilon(1+\beta(1-\epsilon)/3)$*.*

Proof. See Appendix B.1. □

2.7.3 Results for the $(3, 6)$ and $(3, 7)$ base DDs

Fig. 2.15 shows the computed rate-threshold curve parametrized by ν for both the $\mathcal{C}_{3,K,\nu,\beta}$ ensembles, both with $\beta = 0$, i.e., when the code graph has no generalized variable nodes, and with $\beta = 0.3$. We use a $(3, 6)$ base DD with code R-I (see Table 2.3) as component code. While in the former case the minimun gap to capacity is achieved for the base LDPC code ensemble (with a gap to capacity of 0.0710), by using a certain amount of generalized variable nodes we are able to reduce this gap to 0.0592. Further, since all variable nodes in the graph have degree 3, by Lemma 3, for any value of ν the code ensemble has a minimum distance that grows linearly with the block length. Fig. 2.16 shows similar results for a $(3, 7)$ base DD with Code R-III as component code.

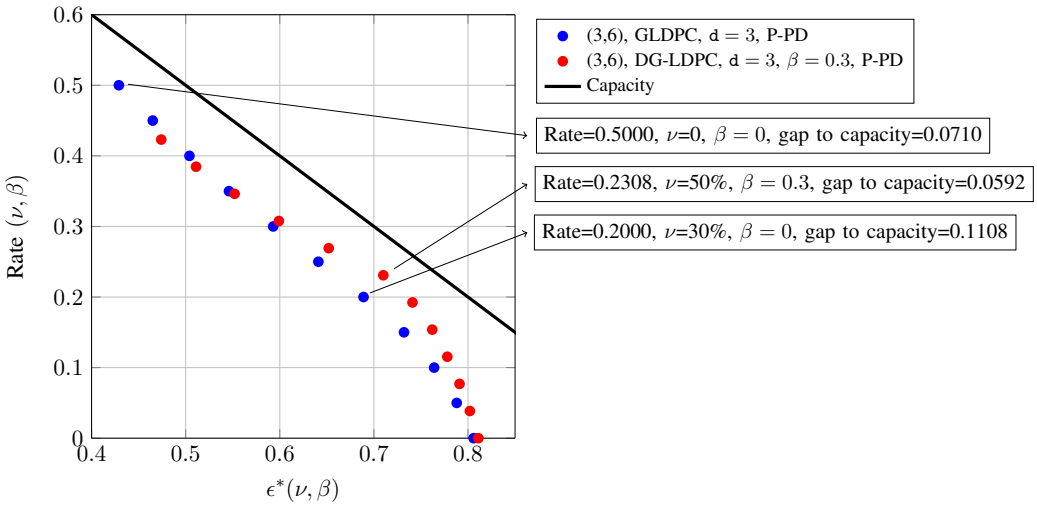


Figure 2.15: $\mathcal{C}_{3,K,\nu,\beta}$ coding rate for a (3,6) base DD with Code R-I as component code, GV nodes constructed using the generator matrix in (2.43), and $\beta = 0.3$.

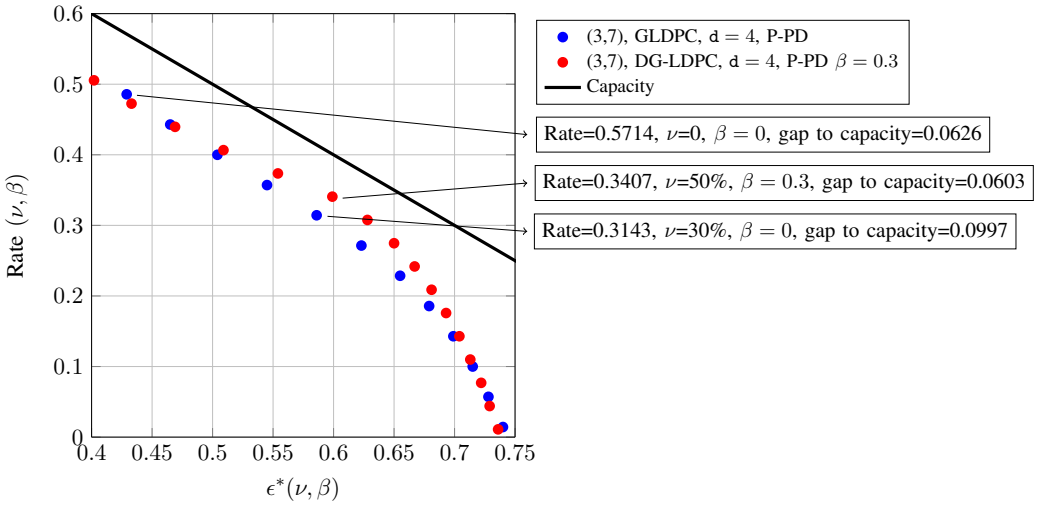


Figure 2.16: $\mathcal{C}_{3,K,\nu,\beta}$ coding rate for (3,7) base DD, Code R-III component code, GV nodes constructed using the generator matrix in (2.43), and $\beta = 0.3$.

2.8 Applications of GLDPC codes

We proposed the P-PD algorithm as a flexible and efficient decoding algorithm that allows us to easily incorporate ML-decoded GC nodes with specific properties into the asymptotic analysis and still maintain a random definition of the graph degree distribution. Using P-PD, asymptotic analysis of the GLDPC ensemble is carried out by a simple generalization of the original PD analysis by Luby et al. in [50]. The only information required about the component code and its decoding method is the fraction of decodable erasure patterns of a certain weight. We consider a class of GLDPC code ensembles characterized by a regular base DD where we include a certain fraction of GC nodes, and we study the tradeoff between iterative decoding threshold, coding rate and minimum distance. We have shown that one can find a fraction of GC nodes required that reduces the original gap to capacity and yields a GLDPC ensemble with linear growth of the minimum distance w.r.t. the block length. Finally, we show how the P-PD analysis can be combined with additional techniques to find a better balance between coding rate and asymptotic gap to capacity. In particular, we consider random puncturing and the use of generalized variable nodes. We would like to emphasize that, in the proposed analysis framework, the evaluation of both coding rate and of iterative decoding threshold are decoupled problems. Consequently, broader classes of component codes or improved decoding methods at GC nodes can be incorporated in a systematic way.

In the next chapter we analyze the GLDPC codes with regular base DD and a certain fraction of GC nodes in the finite-length regime. Due to their regularity of the DD, we show such codes possess a robust finite-length behavior compared to GLDPC code designs proposed in the literature, characterized by capacity-achieving DDs. Furthermore, the simulation results of designed codes outperform other potential candidate codes, which makes our designed codes candidates for ultra reliable low latency communication, such as 5G.

3

Generalized LDPC Codes for Ultra Reliable Low Latency Communication

Fifth-generation (5G) systems aim to increase the capacity of existing mobile networks by a factor of 1000 [15], supporting an extremely high user density, as well as numerous device-to-device and machine communications. Ultra Reliable Low Latency Communication (URLLC) constitutes one of the critical operating regimes in 5G, since it will enable low-cost and power-efficient anywhere and anytime signalling services [80]. The selected channel code must have an excellent error rate performance in a specific range of block lengths and code rates; low computation complexity, low latency, low cost and higher flexibility are also critical [28].

A number of potential candidate codes for 5G URLLC have been proposed recently. A representative summary can be found in three recent papers [28, 92, 90], where, among the coding schemes compared, low-density parity-check (LDPC),

polar codes, and convolutional/turbo codes stand out in the comparisons. To meet the predicted constraints of a host of machine-to-machine (M2M) communications, the authors in [92] consider a low coding rate $R = 1/12$ and short block lengths (480 bits or 2400 bits). A polar code stands out in the performance comparison, although this solution is limited by the decoding delay imposed by the sequential nature of successive cancelation (SC) decoding algorithms, which ultimately limits the decoding throughput. Furthermore, polar code design is channel dependent, hence not versatile for mobile fading channels. In [28], the comparison focuses on larger coding rates, $R = 1/3$, $R = 1/2$, and $R = 2/3$, with similar block lengths to [92]. As in [92], a polar code with SC decoding combined with a cyclic redundancy check (CRC) outperforms turbo and LDPC codes. However, LDPC codes exhibit relatively good performance over all the coding rates and block lengths considered without the aid of a CRC outer code. Similar conclusions are drawn in [90], where the low complexity and high-throughput decoder implementations associated with iterative message passing schemes are emphasized to be desirable in practice.

Our goal in this work is to present GLDPC block codes as a strong candidate for URLLC that, so far, has been largely ignored by the community. We will show that quasi-cyclic GLDPC (QC-GLDPC) codes combined with simple hard-decision decoded outer codes are able to surpass the decoding performance reported in [28, 92] with iterative message passing decoding algorithms. To this end, we propose a novel GLDPC design methodology that has its roots in the contribution presented in Chapter 2. Indeed in Chapter 2, it is shown that the tradeoff between rate and iterative decoding threshold presents a unique optimal operational point where the gap to capacity is minimized as we vary the proportion of GC nodes in the GLDPC code graph. Using a GLDPC code operating at exactly this rate, we first optimize a quasi-cyclic (QC) graph lifting to avoid harmful small structures in the graph. The QC structure also has the benefit of efficient hardware implementation [45, 106, 93, 12] and analysis [13, 25]. The locations of GC nodes are optimized to avoid weak areas in the graph (i.e., many variables connected together using only SPC nodes). As the GLDPC optimal operational rate is typically larger than

the target rate (e.g., $R = 1/12$), we combine the optimized GLDPC code with a complementary low-complexity hard-decision decoded outer code that is designed to match the overall rate to the desired target. Note that the use of a hard-decision decoded outer code allows for flexible and low-complexity rate adaptation. To the best of our knowledge, the idea of combining a GLDPC code with an outer code as a viable solution for low latency 5G URLLC is novel in the area. We note that we do not propose any class of turbo-like decoding scheme, in which the inner (G)LDPC code and the outer code exchange messages iteratively [14, 89, 104, 105]. In our proposal, to limit the complexity, the hard-decision outer code cleans up some of the errors remaining after the GLDPC decoding stage and its decision is not fed back to the GLDPC decoder.

In particular, we propose exemplary designs using (J, K)-regular QC-GLDPC codes with degree-2 variable nodes ($J = 2$), which allow efficient implementation of the GLDPC message passing decoder, since variable nodes only have to propagate (pass) incoming messages without performing any computation. We note that, unlike a conventional LDPC code, a $(2, K)$-regular GLDPC code has good distance properties and message passing performance [47]. We consider schemes with degree $K = 6$, $K = 7$, and $K = 15$ constraint nodes and propose designs that can meet a variety of target coding rate constraints up to $R = 1/2$. To reduce decoding complexity, we propose different sub-optimal decoding algorithms in which we investigate the effect of varying the number of decoding iterations, update rules (including a hybrid min-sum GLDPC decoder), and message quantization. Even with such practical limitations, performance comparisons with the candidates proposed in [28] and [92] show that remarkable error control performance can be achieved over the AWGN channel with quadrature phase shift keying (QPSK) modulation. Ultimately, this paper aims to present general design rules for QC-GLDPC and demonstrate their strength and suitability as candidates for power-constrained devices, such as those in 5G URLLC applications.

The remainder of the chapter is structured as follows: In Section 3.1, we introduce the GLDPC code ensembles and the notation used to characterize the

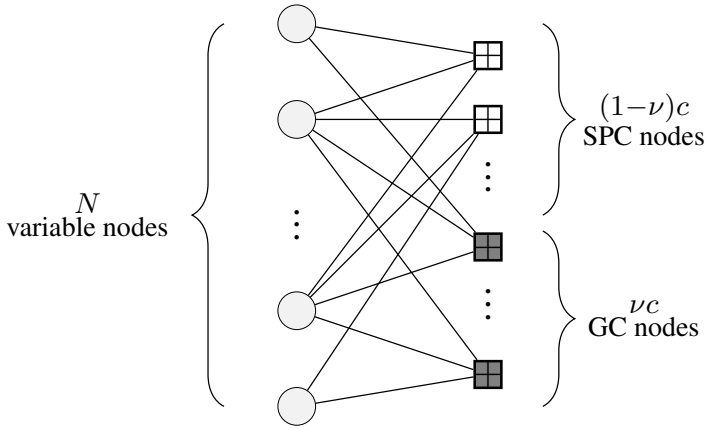


Figure 3.1: Tanner graph of a GLDPC code, where c is the total number of check nodes in the Tanner graph.

degree distribution (DD) of the ensemble. Section 3.2 presents the practical code design process for 5G URLLC. In Section 3.3 we investigate the message passing process of GLDPC codes, including suboptimal message-passing update rules, finite-precision with uniform quantization, the effects of different maximum iteration numbers, and the decoding complexity. In Section 3.4, we compare the performance of the designed code with the codes proposed in [28, 92]. Finally, in Section 4.1 we conclude the paper with a discussion of our results.

3.1 Parameters of the designed GLDPC codes

In this section, we introduce the notation used to define the properties of the GLDPC code ensembles considered in this paper. We restrict our attention to (J, K)-regular graphs, where J is the variable node degree and K is the check node degree, since regular graphs are attractive for VLSI decoder implementation and possess robust finite-length scaling behavior [4]. Following [46], we consider a GLDPC code ensemble that is obtained from an LDPC code ensemble (e.g., an LDPC code ensemble defined by a protograph [97], a QC ensemble, or following a degree distribution $(\lambda(x), \rho(x))$) by replacing a randomly-chosen fraction ν of SPC nodes with identical GC nodes corresponding to an $(K, K-m)$ component code, while the remaining constraint nodes are SPC, where K is the block length

of the component code, $K-m$ is the dimension of the code, and m is the number of rows in the parity check matrix of the linear component code. The Tanner graph of a GLDPC code from such an ensemble with block length N is illustrated in Fig. 3.1. The Tanner graph of any code in this ensemble contains N variable nodes, c check nodes, $\nu\frac{J}{K}N$ GC nodes, and $(1-\nu)\frac{J}{K}N$ SPC nodes. We refer to the LDPC ensemble obtained by taking $\nu = 0$ as the *underlying LDPC code ensemble* or simply the *underlying ensemble*. The design rate of the underlying ensemble $\mathtt{R}_0$ is given as $\mathtt{R}_0 = 1 - J/K$ and the design rate $\mathtt{R}(\nu)$ of the GLDPC ensemble is given by $\mathtt{R}(\nu) = \mathtt{R}_0 - \nu(1-\mathtt{R}_0)(m-1)$. We assume that the incoming edges to every degree-K GC node are assigned uniformly at random to each position of the component code.[1]

In the rest of the paper, as the component code at GC nodes we will present exemplary design results for :

1. $(2, 6)$-regular GLDPC codes with $(6, 3, d_{\min} = 3)$ shortened Hamming linear block codes as GC component codes and generator matrix

$$\mathbf{G}^{(6,3)} = \begin{bmatrix} 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 \end{bmatrix}; \tag{3.1}$$

2. $(2, 7)$-regular GLDPC codes with $(7, 4, d_{\min} = 3)$ Hamming linear block codes as GC component codes; and

3. $(2, 15)$-regular GLDPC codes with $(15, 11, d_{\min} = 3)$ linear block component

[1]Note that the GLDPC ensemble has three sources of randomness: the underlying Tanner (LDPC) graph, the location of the GC nodes in the graph, and the edge labeling at each GC node.

codes with generator matrix

$$\mathbf{G}^{(15,11)} = \begin{bmatrix} 0&1&0&1&0&1&1&0&0&0&0&0&0&0&0 \\ 0&0&0&1&0&1&0&0&1&0&0&0&0&0&0 \\ 0&0&0&1&0&1&0&0&0&1&0&0&0&0&1 \\ 0&0&0&1&0&0&1&0&0&0&1&0&0&0&0 \\ 0&0&0&1&0&0&1&0&0&0&0&1&0&0&1 \\ 0&0&0&0&0&1&1&0&0&0&0&0&1&0&0 \\ 0&0&0&0&0&1&1&0&0&0&0&0&0&1&1 \\ 0&0&0&0&0&0&1&1&0&0&0&0&0&0&1 \\ 0&0&0&0&1&1&0&0&0&0&0&0&0&0&1 \\ 0&0&1&1&0&0&0&0&0&0&0&0&0&0&1 \\ 1&0&0&1&0&1&1&0&0&0&0&0&0&0&1 \end{bmatrix}. \tag{3.2}$$

This code is taken from the database [29, 30], which implements the tools proposed in [8] to design block codes with the largest distance spectrum.

Note that all ensembles only contain degree-2 variable nodes, which allows simpler message passing as a result of their low density. While the $(2, 15)$-regular GLDPC code ensemble better accommodates larger coding rates (roughly up to $\mathtt{R} = 1/2$), the $(2, 6)$ and $(2, 7)$ ensembles have better decoding complexity due to the lower graph density. These two ensembles illustrate the GLDPC complexity/performance trade-offs. Following the design methodology proposed in this paper, we note it may be possible to find GLDPC ensembles that achieve better complexity-performance tradeoffs. For example, as described in Section 3.3.4, the additional decoding complexity of the $(2, 15)$-regular GLDPC is significant due to the high rate component codes. In this regard, as described in [46], alternative regular/irregular GLDPC ensembles with less graph code complexity could be considered.

Using the asymptotic analysis proposed in [46] for a binary erasure channel (BEC), we investigate the tradeoff between rate and the iterative-decoding threshold as a function of ν for the $(2, 6)$-regular, $(2, 7)$-regular and $(2, 15)$-regular GLDPC ensembles. The results are shown in Fig. 3.2. Observe that the asymp-

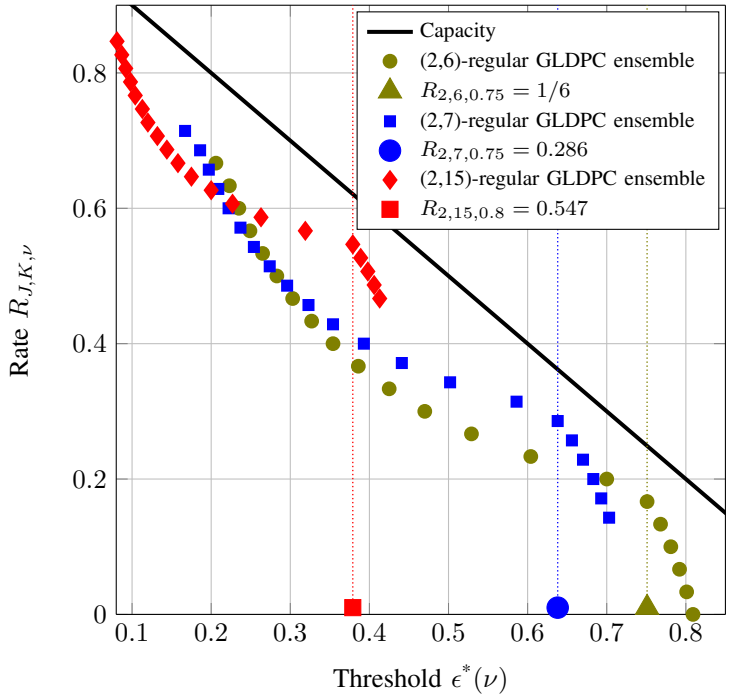


Figure 3.2: Design rate vs. BEC threshold of ensembles of $(2, 6)$-regular, $(2, 7)$-regular, and $(2, 15)$-regular GLDPC codes as a function of the proportion of GC nodes ν in the graph.

totic gap to capacity, which is also a crucial parameter in the finite-length behavior of iteratively-decoded LDPC ensembles [4], is minimized for $\nu = 0.75$ in the $(2, 6)$-regular, and $(2, 7)$-regular cases, and for $\nu = 0.80$ in the $(2, 15)$-regular case. Beyond these values of ν, the gap to capacity increases and, as we will see in the next section, dramatically impacts the finite-length performance of the code. In light of these results, we propose to combine $\mathtt{R}_{2,6,0.75} = 1/6$ $(2, 6)$-regular GLDPC codes, $\mathtt{R}_{2,7,0.75} = 0.286$ $(2, 7)$-regular GLDPC codes, and $\mathtt{R}_{2,15,0.8} = 0.547$ $(2, 15)$-regular GLDPC codes with low-complexity hard-decoded outer codes to match the target coding rate. In particular, the $(2, 6)$-regular GLDPC codes can be used as a component of the concatenated scheme when the target rate is below $\mathtt{R} = 1/6$, the $(2, 7)$-regular GLDPC codes when the target rate is below $\mathtt{R} = 0.286$, and the $(2, 15)$-regular can be used when the target rate is below $\mathtt{R} = 0.547$.

Table 3.1: Design parameters of the proposed concatenated GLDPC coding schemes

Underlying Ensemble	ν	$\mathtt{R}(\nu)$	R_{outer}	Information Length M	Block length N	R
$(2,6)$-regular	0.75	1/6	40/79	40	474	40/474
$(2,7)$-regular	0.75	0.286	40/138	40	469	40/469
$(2,15)$-regular	0.80	0.547	40/254	40	465	40/465
$(2,15)$-regular	0.80	0.547	233/254	233	465	233/465
$(2,15)$-regular	0.80	0.547	155/254	155	465	155/465

3.2 Practical GLDPC Code Design for 5G URLLC

In this section, we investigate several aspects of code design, including QC graph lifting, placement of GC nodes, and outer code design/code rate matching.

3.2.1 Code Design Parameters

In [28] and [92], several coding-rates and block lengths are considered to test coding scheme candidate for URLLC. As exemplary scenarios, we will compare our proposed scheme (QC-GLDPC combined with an appropriate outer code to match the rate) with some of the URLLC candidates in [28] and [92] using the following specifications:

- Overall coding rate $\mathtt{R}$: $\mathtt{R} = 1/12$ [92], $\mathtt{R} = 1/3$, and $\mathtt{R} = 1/2$ [28].
- Information length M: $M = 40$ bits [92], $M = 170$, and $M = 256$ bits [28].
- Block length N: $N = 480$ bits [92], $N = 512$ bits [28].

The design parameters of our proposed schemes are listed in Table 3.1. As we describe in the sequel, the granularity of the QC-structure in the underlying GLDPC graph slightly restricts the design parameters. As a consequence, the proposed coding schemes may not exactly match the above specifications, but we stress that our comparisons will always be fair with the results of [28] and [92] in the sense that our proposed constructions will have slightly larger coding rates and smaller block lengths.

3.2.2 QC Graph Lifting

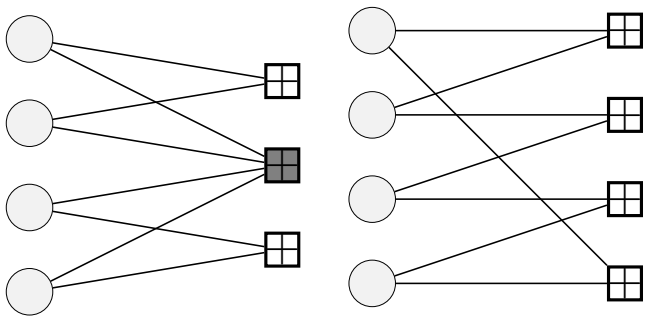
The underlying LDPC code ensemble for a given length N can be drawn from a random ensemble defined by a degree distribution $(\lambda(x), \rho(x))$, from a semi-structured protograph-based ensemble, or from the structured sub-ensemble of QC codes, where the permutation matrices selected in the protograph-based construction are restricted to be circulant. It is well known that the algebraic structure of QC codes allows simple encoding using shift registers, with a complexity linear in the block length [17]. Properly-designed QC graphs have been shown to perform as well as computer-generated random LDPC codes, regular or irregular, in terms of bit-error performance, block-error performance, and error floor for codes with short to moderate block lengths [45].

We first write the parity-check matrix $\mathbf{H}$ of the underlying $(2, K)$-regular QC-LDPC code, lifted from the all-ones base matrix of size $s \times s$ with lifting factor $s_{i,j}$ and code length $N = sK$, as

$$\mathbf{H} = \begin{bmatrix} \mathbf{I}(0) & \mathbf{I}(0) & \cdots & \mathbf{I}(0) \\ \mathbf{I}(0) & \mathbf{I}(s_{1,1}) & \cdots & \mathbf{I}(s_{1,K-1}) \\ \vdots & \vdots & & \vdots \\ \mathbf{I}(0) & \mathbf{I}(s_{J-1,1}) & \cdots & \mathbf{I}(s_{J-1,K-1}) \end{bmatrix}, \tag{3.3}$$

where $s_{i,j}, 1 \leq i \leq J-1, 1 \leq j \leq K-1$ are the *left shifting parameters*, such that $\mathbf{I}(0)$ is the $s \times s$ identity matrix and $\mathbf{I}(s_{i,j})$ is the left shifted $s \times s$ identity matrix, where each row of $\mathbf{I}(0)$ is circularly shifted to the left by $s_{i,j}$ positions.

In order to guide our design, we randomly sampled 100 codes from the $(2, 6)$-regular QC GLDPC ensemble with block length 474 (using a random placement of the fraction $\nu = 0.75$ of GC nodes in the graph) and empirically determined the dominant error objects over the BEC at moderate to high SNRs. The two structures that were found to dominate the code performance in the error floor region are shown in Fig. 3.3. Fig. 3.3(a) corresponds to two 4-cycles connected by a GC node and Fig. 3.3(b) corresponds to an 8-cycle composed of SPC nodes. Both of these objects, and some other less dominant objects not shown here, can be eliminated simply by increasing the girth g of the base LDPC graph.



(a) (b)

Figure 3.3: Dominant error patterns detected in randomly constructed $(2, 6)$-regular QC GLDPC codes. Gray shaded squares represent GC nodes, while white squares represent SPC nodes.

Following [25], to ensure that the $\mathbf{H}$ matrix defined in (3.3) has a girth of at least $2(i + 1)$, a necessary and sufficient condition is

$$\sum_{t=0}^{m-1} \Delta_{j_t,j_{t+1}}(k_t) \neq 0 \bmod s, \tag{3.4}$$

where $\Delta_{j_t,j_{t+1}}(k_t) = s_{j_t,k_t} - s_{j_{t+1},k_t}$ for all $2 \leq m \leq i$, $0 \leq j_t, j_{t+1} \leq J - 1$ and $0 \leq k_t \leq K - 1$, with $j_0 = j_m$, $j_t \neq j_{t+1}$, and $k_t \neq k_{t+1}$.

Design of the $(2, 6)$-regular Underlying QC Graph

Given that all of the check nodes have degree 6 and that s should be chosen to be a prime, we selected $s = 79$, which gives a slightly smaller block length of 474 bits than the 480 bits used in [92]. The resulting $(2, 6)$-regular matrix has the form

$$\mathbf{H}_{(2,6)} = \begin{bmatrix} \mathbf{I}(0) & \mathbf{I}(0) & \mathbf{I}(0) & \mathbf{I}(0) & \mathbf{I}(0) & \mathbf{I}(0) \\ \mathbf{I}(0) & \mathbf{I}(s_{1,1}) & \mathbf{I}(s_{1,2}) & \mathbf{I}(s_{1,3}) & \mathbf{I}(s_{1,4}) & \mathbf{I}(s_{1,5}) \end{bmatrix}. \tag{3.5}$$

There are many possible ways of choosing $s_{1,j}, 1 \leq j \leq 5$, to maximum girth and/or improve code performance. We remark at this point that girth optimization is greatly facilitated by the low-density $(2, 6)$-regular structure. Note that $(2, 6)$-regular LDPC codes have poor distance properties. In fact, any QC-LDPC code of the form (3.5) has $d_{min} \leq 6$, independent of s [54]. This implies, in turn, that

the largest girth we can achieve is $g = 12$ in the base LDPC code, since a cycle of length $2c$ implies the existence of a codeword of weight c in a $(2, K)$-regular code. In order to choose the shift parameters, we can make use of Theorem 2.1 in [25], and adopt the so-called *Power* construction to select $s_{1,j}$, $1 \leq j \leq 5$ as $s_{1,j} = a^1 b^j \bmod s$, which promises $g \leq 12$. For $s = 79$, if we choose $a = 3$, and $b = 5$, this gives $[s_{1,1}, s_{1,2}, s_{1,3}, s_{1,4}, s_{1,5}] = [15, 75, 59, 58, 53]$ and achieves $g = 6$. Alternatively, since the shift parameter space is relatively small for the $(2, 6)$-ensemble, we could run a straightforward search, which provides a QC graph with girth $g = 12$ and shift parameters

$$[s_{1,1}, s_{1,2}, s_{1,3}, s_{1,4}, s_{1,5}] = [54, 66, 71, 55, 69]. \tag{3.6}$$

Design of the $(2, 7)$-regular Underlying QC Graph

Similar to the $(2, 6)$-regular case, we pick $s = 67$ and use the power construction with $a = 3$, and $b = 5$. This gives $[s_{1,1}, s_{1,2}, s_{1,3}, s_{1,4}, s_{1,5}] = [15, 4, 20, 29, 3, 15]$ with resulting girth $g = 6$. Given that the shift parameter space is still relatively small, we can also run an exhaustive search, which provided a QC graph with girth $g = 12$ and shift parameters

$$[s_{1,1}, s_{1,2}, s_{1,3}, s_{1,4}, s_{1,5}, s_{1,6}] = [61, 49, 44, 1, 46, 14]. \tag{3.7}$$

Design of the $(2, 15)$-regular Underlying QC Graph

For the $(2, 15)$ case, we proceed similarly and pick $s = 31$, which results in a block length of $N = 465$. However, a brute-force search is more complicated in this case due to the dimension of the shift parameter space. Applying the Power construction to find suitable shift parameters, we obtain a QC girth with the largest possible girth, $g = 12$, by using $a = 2$ and $b = 3$, where

$$\begin{aligned}&[s_{1,1}, s_{1,2}, \ldots, s_{1,14}] \\ &\quad = [6, 18, 23, 7, 21, 1, 3, 9, 27, 19, 28, 16, 17, 20].\end{aligned} \tag{3.8}$$

GLDPC performance with QC underlying LDPC graphs

In Fig. 3.4, we plot the GLDPC bit error rate performance obtained on the BEC of several $(2, 6)$-regular, $(2, 7)$-regular, and $(2, 15)$-regular underlying LDPC code ensembles, including QC-LDPC codes constructed using the shift parameter set in (3.6), (3.7) and (3.8) all with girth 12. The $(2, 6)$-GLDPC code and the $(2, 7)$-GLDPC code have $\nu = 75\%$ of GC nodes in the graph, while the $(2, 15)$-GLDPC code ensemble has $\nu = 80\%$ GC nodes in the graph. We also include the GLDPC performance when we use underlying QC-LDPC codes constructed following the power method with $[a, b] = [3, 5]$ (girth $g = 6$), unstructured randomly constructed graph codes, and randomly constructed semi-structured protograph-based codes (but not QC). In ll simulations, we set the maximum number of allowed decoding iterations to $I_{\max} = 50$. We remark that the waterfall performance of all codes of a given rate are similar, but the GLDPC error floor is optimized for the QC-LDPC underlying codes with the proposed QC designs, which demonstrates that the robustness against error floor is inherited by the code after a certain fraction of SPC nodes are replaced by GC nodes (recall that our earlier motivation was to increase the girth to remove harmful objects from the graphs of the GLDPC codes). Finally, note that the $(2, 15)$-regular QC-LDPC code displays a steeper error rate decrease, potentially giving rise to a lower error floor.

3.2.3 Location of the GC Nodes

After the underlying QC-LDPC code graph is designed, we turn our attention to the locations of the GC nodes. As discussed in Section 3.1, the optimal proportion, from a threshold perspective, is that 75% percent of the check nodes should be replaced by GC nodes in the $(2, 6)$-regular and $(2, 7)$-regular cases. This fraction increases to the 80% in the $(2, 15)$-regular case. In Fig. 3.5, we show the average performance (red circles) of the proposed rate $R = 1/12$ $(2, 6)$-regular QC-GLDPC code, including a rate $R_{\text{outer}} = 40/79$ outer code (see Table 3.1) that corrects up to 15 errors,[2] obtained over 600 randomly chosen GC node locations (all using the

[2] As described in Section 3.2.4, this is rather a conservative assumption.

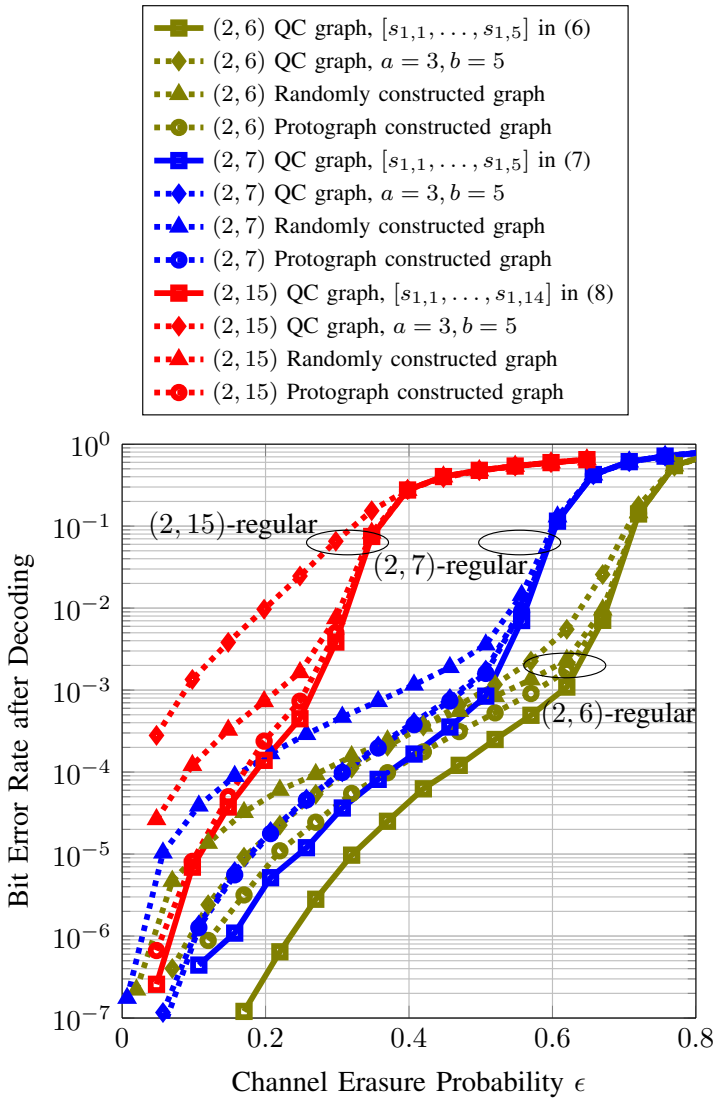


Figure 3.4: Bit error rate for the BEC for a selection of $(2, 6)$-regular, $(2, 7)$-regular and $(2, 15)$-regular GLDPC codes with different underlying graph constructions. The $(2, 6)$-GLDPC code and the $(2, 7)$-GLDPC code have $\nu = 75\%$ of GC nodes in the graph, while the $(2, 15)$-GLDPC code ensemble has $\nu = 80\%$ GC nodes in the graph.

same underlying QC graph). The GLDPC message passing decoder (see Section 3.3) is run for 5 iterations. We also highlight the best performing case (blue triangles), which achieves a gain of 0.2 dB over the average at a BLER equal to 10^{-5}. We further include the performance of a GLDPC code with a hand-crafted location of GC nodes (black squares) that are intended to give poor performance by ensuring that the remaining SPC nodes in the graph are all connected to the same set of variable nodes, thereby creating a weak region in the GLDPC graph and resulting in significant performance loss. With such a large fraction of GC nodes in the graph, the performance of the resulting GLDPC code is reasonably robust with respect to the locations of the GC nodes in the graph, unless we specifically create regions of the graph with multiple local SPC nodes are connected to the same set of variables. However, there is likely to be a larger variance in performance for smaller fraction of GC nodes.

3.2.4 Target Coding Rates

To adapt the designed GLDPC code to other target rates, such as those in the 5G URLLC regime, we consider techniques to lower the coding rate and improve the error correcting capability. Among others, this could be achieved by adding more GC nodes and/or utilizing an outer code. (Similarly, the rate could be increased by using fewer GC nodes and/or puncturing.) Both approaches have advantages and disadvantages. From Fig. 3.2, we observe that the gap to capacity grows as we move away from the optimal operational point of the given GLDPC ensemble, i.e., if we decrease the GLDPC coding rate $R_{J,K,\nu}$ by including a larger fraction of GC nodes in the graph. This will certainly impact the GLDPC finite-length performance, since it is well known that the gap to capacity is one of the critical parameters of the finite-length scaling law of iteratively decoded LDPC code ensembles [4]. As an alternative, we propose to maintain the GLDPC coding rate at its optimal point ν (from an asymptotic perspective), i.e., use a rate $R_{2,6,0.75} = 1/6$ $(2, 6)$-regular GLDPC code with $\nu = 0.75$, a rate $R_{2,7,0.75} = 0.286$ $(2, 7)$-regular GLDPC code with $\nu = 0.75$, or a rate $R_{2,15,0.80} = 0.547$ $(2, 15)$-regular GLDPC code with

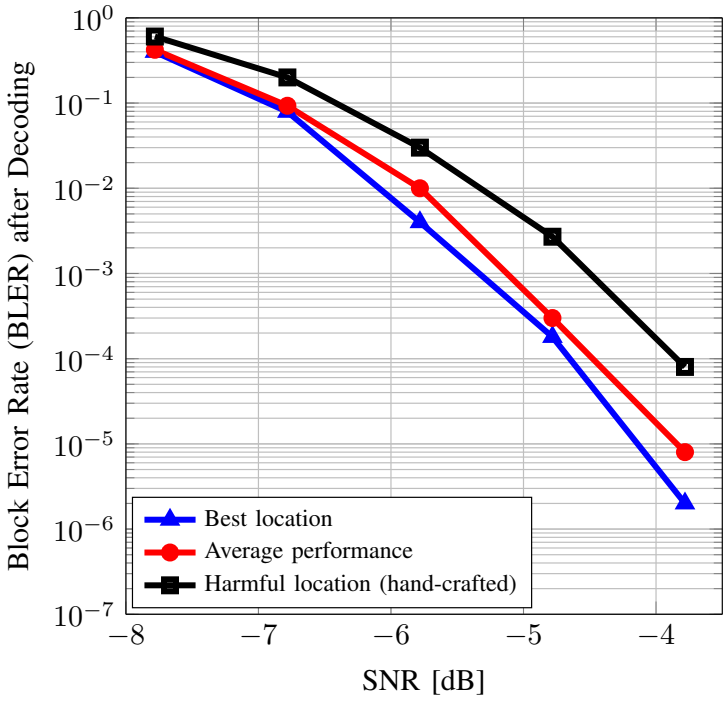


Figure 3.5: BLER over an AWGN channel with QPSK modulation for the proposed $(2, 6)$-regular GLDPC coding scheme with different locations of GC nodes in the graph. Results were obtained using a rate $R_{\text{outer}} = 40/79$ outer code (see Table 3.1) that corrects up to 15 errors.

$\nu = 0.8$, and lower the rate accordingly by using a rate R_{outer} low-complexity hard-decoded outer code.

As a representative comparison, in Fig. 3.6 we compare the BLER performance of a rate $R_{2,6,0.875} = 40/474 \approx 1/12$ GLDPC code, obtained by selecting $\nu = 0.875$, versus that of the rate $R_{2,6,0.75} = 1/6$ GLDPC code with a rate $R_{\text{outer}} = 40/79$ outer code (resulting in approximately the same overall coding rate $R = 1/12$, see Table 3.1). Note that we assume a systematic generator matrix. In all our simulations, the outer code is applied with a genie over the whole block, i.e., we correct up to a certain amount of errors over the whole block assuming a worst-case scenario that those bits are all information bits. Results with an actual implementation of the systematic scheme can only be better. The outer code can be chosen to be any (n, k) linear block code of appropriate length and rate to

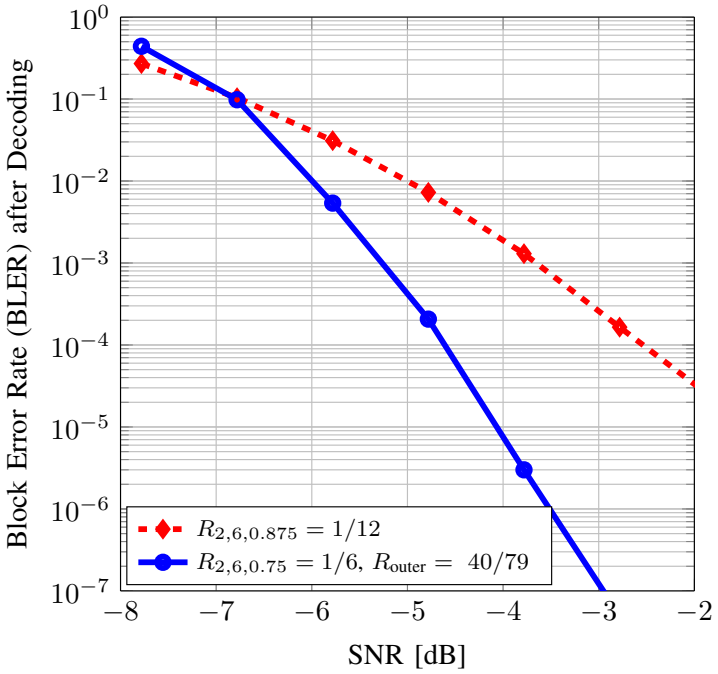


Figure 3.6: BLER on an AWGN channel with QPSK modulation as a function of the SNR for a rate $R_{2,6,0.875} = 40/474 \approx 1/12$ GLDPC code and a $\mathtt{R} = 1/12$ scheme that combines a rate $R_{2,6,0.75} = 1/6$ GLDPC code with $R_{\text{outer}} = 40/79$ outer code that can correct up to 15 errors.

meet the target, such as a $(79, 40)$ shortened BCH code. We would expect to use a low-cost, high-speed, hard-decision decodable code for the outer code. With a block length of 79 bits and a rate $R_{\text{outer}} = 40/79$, we can conservatively assume that the outer decoder can correct up to 15 errors [17]. For this comparison, both $(2, 6)$-regular GLDPC codes were randomly constructed following the protograph method with randomly placed GC nodes (similar results are obtained for different random draws of the matrices). The GLDPC message passing decoder (see Section 3.3) is run for at most $I_{\max} = 5$ iterations in both cases. The results show that the higher rate GLDPC code, optimized for threshold, with a hard-decision outer code has significantly better performance than the GLDPC code alone that was constructed by adding more GC nodes. Note that, in addition to good waterfall performance, we do not observe an error floor down to a BLER of 10^{-8} with the outer code version.

3.3 GLDPC Message Passing

In this section, we discuss the message passing update decoding rules for the iteratively decodable GLDPC code. Compared to the conventional belief propagation update rules for LDPC decoders, the only difference here is how to process probabilistic messages at the GC nodes. In this regard, the processing depends on the chosen component code. We take the $(2, 6)$-regular GLDPC code as a running example in this section. As described in Section 3.1, the component code used at GC nodes is a shortened $(6, 3)$ Hamming code, with generator matrix given in (3.1) and codebook $\mathcal{C}^{(6,3)}$ written in matrix form as

$$\mathbf{C}^{(6,3)} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 \\ 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 \end{bmatrix}. \tag{3.9}$$

The GLDPC update rule at GC nodes is determined by the component codebook $\mathbf{C}^{(6,3)}$. Let Λ_j denote the input LLR message coming from the j-th variable node connected to the GC node, where index j, $j = 1, 2, \ldots, 6$, corresponds to the jth input to the component code. Let $\tilde{\Lambda}_j$ denote the output LLR message to be sent to the j-th variable node. In Appendix D.1, we show that $\tilde{\Lambda}_j$, $j = 1, 2, \ldots, 6$, can be computed as follows

$$\begin{aligned} \tilde{\Lambda}_j = &\log \left[\sum_{\substack{i \in \{1,8\} \\ C_{i,j}=0}} \exp \left(\sum_{\substack{m \in \{1,6\} \\ m \neq j}} \mathbb{I}[(\mathbf{C}^{(6,3)}_{i,m} = 0)](\Lambda_{pj} - \Lambda^*) \right) \right] \\ &- \log \left[\sum_{\substack{i \in \{1,8\} \\ C_{i,j}=1}} \exp \left(\sum_{\substack{m \in \{1,6\} \\ m \neq j}} \mathbb{I}[(\mathbf{C}^{(6,3)}_{i,m} = 1)](\Lambda_{pj} - \Lambda^*) \right) \right], \end{aligned} \tag{3.10}$$

where $\mathbb{I}[\cdot]$ denotes the indicator function, $\mathbf{C}_{i,m}^{(6,3)}$ denotes the m-th bit of the i-th codeword, $i = 1, 2, \ldots, 8$, $\Lambda^* = \max_j \Lambda_j$, and we use the *log-sum-exp trick* to avoid numerical issues in the evaluation of the exponential terms.[3] Λ^* can be efficiently computed using a digital comparator. Note that at variable nodes and SPC nodes the message passing update rules used are those for standard LDPC decoding [40]. Hereafter, we refer to the update rule in (3.10) as the sum-product algorithm (SPA) GLDPC decoder. Also, in the following we denote the maximum number of message passing iterations as $I_{\max}$. A hard-decision stopping rule is implemented so the decoding terminates when all parity check conditions (at both SPC and GC nodes) are satisfied.

3.3.1 Min-sum Decoding Algorithms

It is well known that floating-point operations such as $\log(\cdot)$ or $\exp(\cdot)$ increase the decoder implementation complexity and its power-consumption [39, 42]. In the following we explore several simplifications of the SPA update rules in (3.10) and investigate the effect on decoder performance. First, we adapt the min-sum decoding algorithm for LDPC decoders [40] to the GC node update rules as

$$\tilde{\Lambda}_j = \max_{\substack{i\in\{1,8\} \\ C_{i,j}=0}} \left(\sum_{\substack{m\in\{1,6\} \\ m\neq j}} \mathbb{I}[(\mathbf{C}_{i,m}^{(6,3)} = 0)](\Lambda_{pj} - \Lambda^*) \right) - \max_{\substack{i\in\{1,8\} \\ C_{i,j}=1}} \left(\sum_{\substack{m\in\{1,6\} \\ m\neq j}} \mathbb{I}[(\mathbf{C}_{i,m}^{(6,3)} = 1)](\Lambda_{pj} - \Lambda^*) \right). \tag{3.11}$$

Comparing (3.10) and (3.11), we have replaced the $\log(\cdot)$ and $\exp(\cdot)$ operators by a simpler maximum-search operator that can be efficiently implemented with a digital comparator. The decoding algorithm based on these update rules is referred hereafter as the min-sum algorithm (MSA) GLDPC decoder. In Fig. 3.7, we compare the performance of the $(2, 6)$-regular GLDPC code with the $R_{\text{outer}} =$

[3] The *log-sum-exp trick* works as follows: let $\mathbf{a} = [a_1, a_2, \ldots, a_d]$ be a real-valued vector. Instead of directly evaluating $b = \log(\sum_{i=1}^{d} \exp(a_d))$, we first compute $a^* = \max_i a_i$ and then we compute b as $b = a^* + \log(\sum_{i=1}^{d} \exp(a_d - a^*))$.

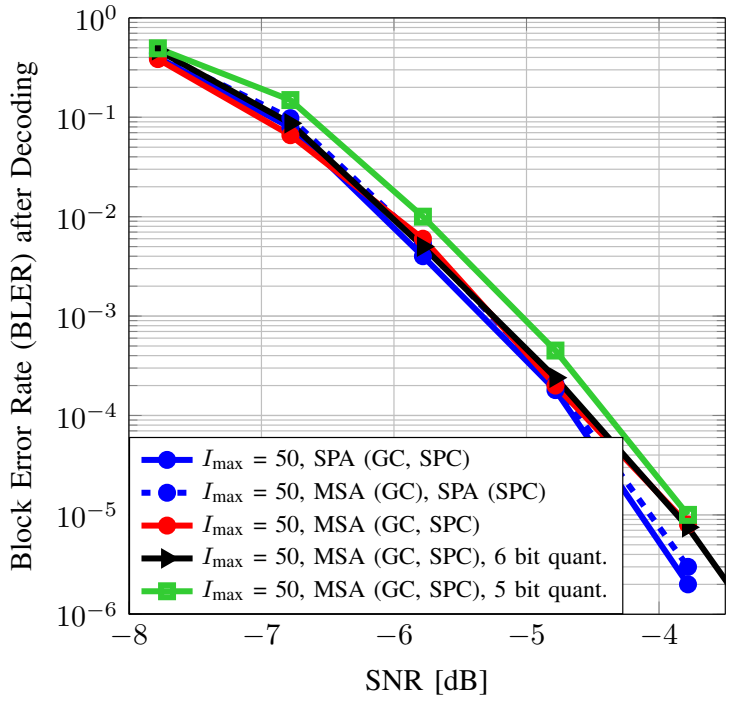


Figure 3.7: BLER over an AWGN channel with QPSK modulation for the the $(2, 6)$-regular GLDPC code with the $R_{\text{outer}} = 40/79$ outer code (see Table 3.1) and different decoding algorithms.

40/79 outer code (see Table 3.1) achieved by the full-precision SPA decoder in (3.10) with the MSA decoder in (3.11). Several combinations of SPA and MSA update rules at either SPC nodes and/or GC nodes are considered. In all cases, we consider a relatively large maximum number of message passing iterations ($I_{\max} = 50$) so that we can assume in all cases that the decoder has been run until convergence. The performance loss is only numerically relevant at large SNR, where the MSA decoder at both GC and SPC nodes loses approximately 0.3 dB at a BLER equal to 10^{-5} compared to full precision SPA at both types of nodes. We note that it is well known that *offset min-sum* and *scaled min-sum* algorithms have been shown to suffer very little to no performance loss when compared to SPA. We have not investigated such improvements to (3.11) in this paper, leaving it for future work.

3.3.2 Finite-precision with Uniform Quantization

Along with low complexity MSA decoding update rules, it is practically relevant to study the effects of quantizing the messages with finite-precision. Here, we consider uniform quantization since it is widely used in practice [72]. We use 1 bit to encode the LLR sign. After analyzing the empirical LLR distribution using $5 \cdot 10^6$ samples at different SNR values (we used SNR $\in [-8, -7, -6, -5, -4]$ dB), we observed that 95% of the distribution was contained in the $[-4, 4]$ range, so 2 bits is considered sufficient to represent the integer part of each LLR message. Finally, in Fig. 3.8 we show the 8 bin empirical histogram of the decimal part of the LLR floating-point messages. Despite more mass being concentrated at small values, the histogram shows a heavy tail of the distribution. Therefore, even though a non-uniform quantizer could allow more precision at small values of the decimal part, it is expected that a 3 bit uniform quantizer would provide an acceptable reconstruction. To verify our expectation, we also include the quantized MS decoder performance when $I_{\max} = 50$ using both 6 bits (1 sign + 2 integer + 3 decimal) and 5 bits (1 sign + 2 integer + 2 decimal) for LLR quantization in Fig. 3.7. Observe that, compared to full precision MSA decoding, the performance loss with 6 bits is numerically negligible.

3.3.3 The Effects of Different Maximum Iteration Numbers

The number of decoding iterations is usually limited in practice since it determines the latency and throughput of the system and heavily impacts the energy consumption of the decoding circuitry [18]. In Fig. 3.9 we evaluate the robustness of the different decoding algorithms considered as we reduce the number $I_{\max}$ of allowed iterations. Curves with circle markers denote full-precision SPA decoding, where we can observe that an approximate 0.25 dB loss at a BLER of 10^{-4} is incurred when $I_{\max}$ is set to only 5 iterations compared to $I_{\max} = 50$. When full precision MSA (square markers) and the same number of iterations is used, this loss is essentially doubled, close to 0.6 dB at a BLER of 10^{-4} . The loss increases up to approximately 1 dB when quantized MSA (triangle markers) is

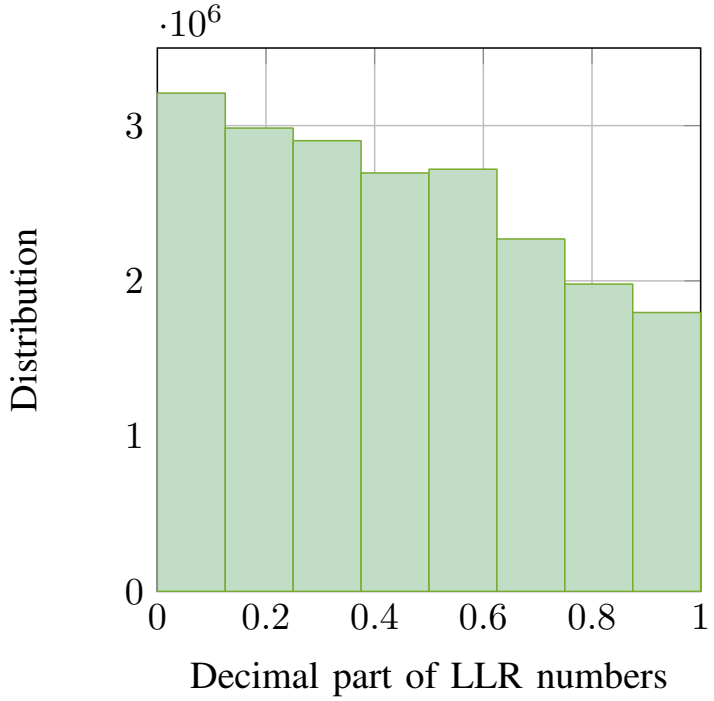


Figure 3.8: Empirically determined histogram of the decimal part of LLR numbers computed for the $(2, 6)$-regular GLDPC code.

considered. Therefore, suboptimal decoding rules may require a larger number of (less complex) iterations to maintain acceptable performance. With $I_{\max} = 10$ the performance loss of each case (when compared to the $I_{\max} = 50$ case) is essentially reduced to half that of $I_{\max} = 5$.

It is important to note that the performance degradation reported is comparable to that reported for non-generalized LDPC codes with a standard MSA decoder [110]. Therefore, it is expected that better complexity/performance tradeoffs can be achieved if more robust implementations of MSA decoding strategies (e.g., attenuated MSA, offset MSA, approximated MSA) are implemented. An in-depth survey of these methods can be found in Chapter 5 of [88].

3.3.4 Decoding Complexity

For the sake of complexity comparison with other URLLC coding schemes proposed in the literature, particularly those in [28, 92], we now determine the computa-

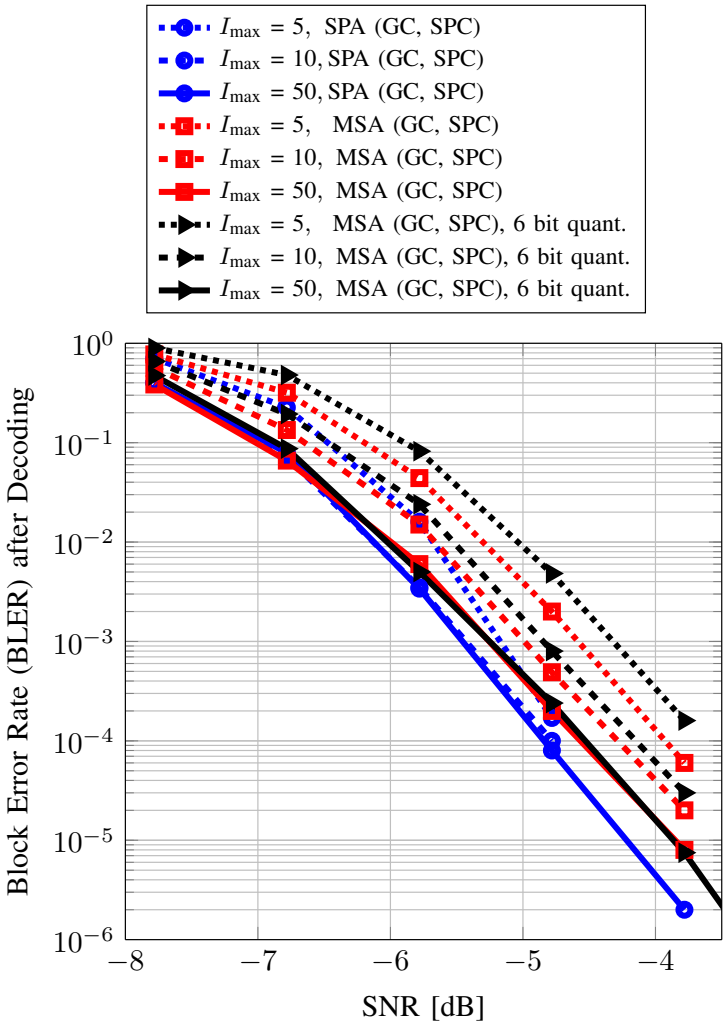


Figure 3.9: BLER over an AWGN channel with QPSK modulation for the $(2, 6)$-regular GLDPC code with an $R_{\text{outer}} = 40/79$ outer code (see Table 3.1) and different decoding algorithms.

tional complexity of the GLDPC decoder by enumerating the number of additions, subtractions, multiplications, divisions, comparisons, max (min) operations, and look-up table operations. Most of these operations correspond to one equivalent addition, whereas the comparison operation, in most cases, corresponds to two equivalent additions [92]. In the following, we ignore the hard-decision decoding complexity of the outer code, as the additional complexity is negligible compared to the GLDPC message passing complexity. Also, note that this study does not

differentiate between floating point operations (such as those in a SPA decoder) and the simpler operations required by a MSA decoder. It is nonetheless informative to compare with the results in [28, 92], as the authors there used the same metrics for complexity.

For the $(2, 6)$-regular GLDPC code, according to (3.10) the update of every GC node requires $19 \times K$ additions/subtractions. Also, the SPA update at SPC nodes requires $10 \times K$ multiplications/divisions [40]. Furthermore, note that the variable node degree is $J = 2$, hence there is only one addition to perform when updating the variable nodes, and thus the decoding complexity per iteration for variable node is $J \times N = 948$. Altogether, the decoding complexity per iteration is $J\frac{N}{K}\nu 19K + J\frac{N}{K}(1-\nu)10K + JN = JN(11+9\nu) = 16827$, given $J = 2, N = 474$, and $\nu = 0.75$. If $I_{\max}$ denotes the maximum iteration number, the decoding complexity (in the worst case) is $16827 \times I_{\max}$. Similarly, for the $(2, 7)$-regular GLDPC code, we obtain $48 \times K$ additions/subtractions to update the output messages at every GC node, and $12 \times K$ multiplications/divisions to output messages at every SPC node, and 994 additions to update variable nodes. Thus, the decoding complexity per iteration is $J\frac{N}{K}\nu 48K + J\frac{N}{K}(1-\nu)12K + JN = JN(13 + 36\nu) = 37520$, given $J = 2, N = 469$, and $\nu = 0.75$. The decoding complexity (in the worst case) is $I_{\max} \times 187600$. Finally, by following a similar procedure, we can show that the worst-case complexity for the $(2, 15)$-regular GLDPC code is $I_{\max} \times 10671378$.

For the case $R = 1/12$ with $M = 40$ information length, in Table 3.2 we include a complexity comparison with different coding schemes proposed in [92]. Recall that the GLDPC decoding complexity is dominated by the GC update rule in (3.10), which requires a full enumeration over the component code codebook. The small codebook of the shortened $(6, 3)$ Hamming code in (3.1) and the $(7, 4)$ Hamming code explain the comparable complexity of the $(2, 6)$-regular and $(2, 7)$-regular GLDPC code w.r.t. to the coding schemes in [92]. However, the $(15, 11)$ component code given by (3.2) spans a codebook of size 2048, which explains the near 1000 times complexity factor in Table 3.2. As it is shown in the next section, both GLDPC coding schemes provide remarkable performance gains, even up to

Table 3.2: Decoding complexity of $R = 1/12$ coding schemes

Coding Scheme	Block length	Complexity	Multiplicative factor (w.r.t turbo code)
Turbo code with BCJR decoding in [92]	480	61440.00	1.0000
LDPC with MSA decoding in [92]	480	61880.09	1.0070
Polar code with SCL decoding in [92]	480	61751.19	1.0050
Convolutional code with BCJR decoding in [92]	480	40960.00	0.6670
$(2, 6)$-regular GLDPC, $I_{\max} = 5$	474	84135.00	1.3694
$(2, 7)$-regular GLDPC, $I_{\max} = 5$	469	187600.00	3.0534
$(2, 15)$-regular GLDPC $I_{\max} = 5$	465	53356890.00	868.4390

1 dB or more, at different rates and target error probabilities.

Several options can be explored to find GLDPC code ensembles with better performance/complexity tradeoffs that are also able to scale to larger coding rates. For example, one could explore existing algorithms in the literature that perform approximate soft-decoding of algebraic codes at polynomial cost [69, 6, 37]. This would dramatically reduce the complexity of the $(2, 15)$-regular GLDPC code, for instance, by using $(15, 11)$ Hamming codes instead of the code in (3.2). Alternatively, less dense regular/irregular GLDPC ensembles can be used, such as those investigated in [46], where the asymptotic performance of some quasi-regular GLDPC ensembles were analyzed, along with puncturing to adapt to larger coding rates. An exhaustive analysis of all this possible design alternatives, including the effect of sub-optimal soft-decoding methods at GC nodes, is beyond the scope of this chapter, in which our main goal is to bring attention to the remarkable performance that GLDPC codes achieve in the short finite-length regime and highlight their potential for practical URLLC applications in 5G and beyond.

3.4 Experimental Results

We now compare the BLER performance over the AWGN channel of the overall rate $R = 1/12, 1/2$, and $1/3$ coding schemes summarized in Table 3.1 versus those with same rates proposed in [28, 92], which include turbo codes with BCJR decoding, LDPC codes with MSA decoding and offset MSA decoding, polar codes with successive cancellation list (SCL) decoding and a CRC outer code, and convolu-

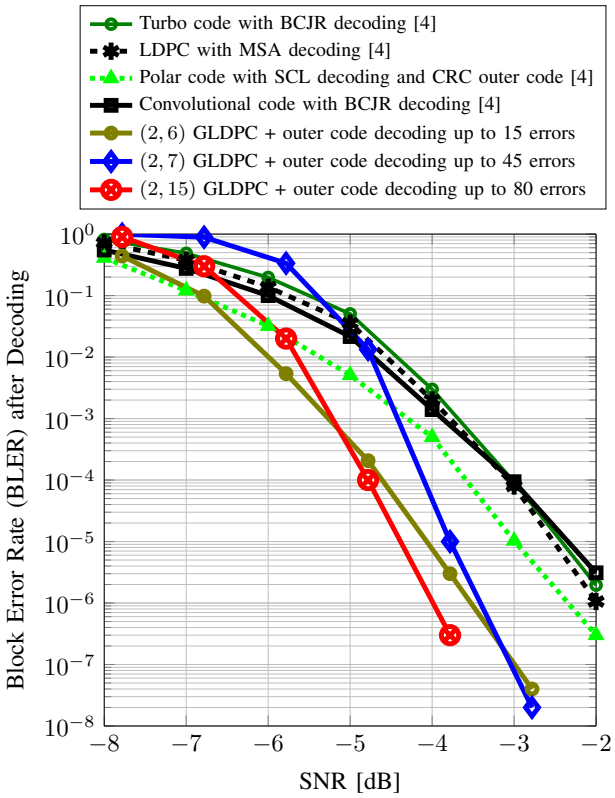


Figure 3.10: BLER over an AWGN channel with QPSK modulation for the proposed coding schemes (first two rows of Table 3.1) compared to several other rate $R = 1/12$ coding schemes with $M = 40$ information bits proposed in [92].

tional codes with BCJR decoding, among others. Specific details on these coding schemes are given [28, 92]. Here we just reproduce their simulation results for the sake of comparison.

To simulate the GLDPC coding scheme, we apply the usual hard-decision syndrome stopping rule for the decoder and, as discussed in Section 3.2.4, we assume a worst scenario where all the remaining errors after GLDPC coding coincide with information bits of a systematic-encoded outer code. The GLDPC message passing is run for up to $I_{\max} = 5$ iterations with full-precision SPA decoding. Fig. 3.10 shows the performance of rate $R = 1/12 \approx 0.0834$ coding schemes proposed in [92] with $M = 40$ information bits and a block length of $N = 480$ bits. We include the performance of the following GLDPC designs (first three rows in Table 3.1):

- Rate $R = 40/474 \approx 0.0844$ $(2, 6)$-regular QC-GLDPC code with a rate $R_{\text{outer}} = 40/79$ outer code (see the first row of Table 3.1), resulting in a block length $N = 474$. We conservatively assume that the outer code can decode up to 15 errors.

- Rate $R = 40/469 \approx 0.0853$ $(2, 7)$-regular QC-GLDPC code with a rate $R_{\text{outer}} = 40/138$ outer code (see the second row of Table 3.1), resulting in a block length $N = 469$. We conservatively assume that the outer code can decode up to 45 errors.

- Rate $R = 40/465 \approx 0.0860$ $(2, 15)$-regular QC-GLDPC code with a rate $R_{\text{outer}} = 40/254$ outer code (see the third row of Table 3.1), for which we again conservatively assume that it can correct up to 80 errors [17].

In all cases, the performance gain is a 1.5 dB at a BLER equal to 10^{-5} over the SCL decoded polar code. The $(2, 15)$-regular QC-GLDPC code achieves even a larger gain at the cost of increased complexity (see Section 3.3.4). The $(2, 7)$-regular QC-GLDPC code outperforms the $(2, 6)$-regular ensemble for high SNRs with a similar complexity, and is less than 1 dB away from the $(2, 15)$-regular QC-GLDPC code at a BLER equal to 10^{-5} where the decoding complexity is roughly three orders of magnitude smaller.

In Fig. 3.11, we extend the analysis to higher rates and we compare our proposed schemes with those in [28]. In Fig. 3.11(a), we consider the rate $R = 233/465 \approx 0.5010$ $(2, 15)$-regular QC-GLDPC code with a rate $R_{\text{outer}} = 233/254$ outer code coding scheme (third row of Table 3.1) and block length $N = 465$. Its BLER performance is compared with codes of slightly lower rate $R = 1/2$ and larger block length $N = 512$ bits. At low-to-moderate SNR values, the performance gain achieved is remarkable, despite that in this case (probably related to the high rate outer code used) it appears to vanish with increasing the SNR. In Fig. 3.11(b), we consider the rate $R = 155/465$ $(2, 15)$-regular GLDPC coding scheme (last row of Table 3.1), and we compare its performance with rate $R = 1/3$ coding schemes with $M = 170$ information bits and block length $N = 512$ proposed in [28]. The $(2, 15)$-regular GLDPC code achieves a gain w.r.t. to state-of-the-art of almost 1.5 dB at BLER of 10^{-4}. In Fig. 3.11(b), we also include a $(2, 7)$-regular GLDPC coding scheme designed to match the target coding rate of $1/3$, in which the fraction of GC nodes in the graph is slightly reduced compared to the optimal fraction $\nu = 75\%$ that yield ed $R \approx 0.286$. By setting $\nu = 0.667$, the coding rate is slightly above $1/3$ ($R = 0.335$). In Fig. 3.11(b) we also show the performance of such a design using the $(2, 7)$-regular QC graph with $N = 469$ and no outer code. Observe that, despite the pronounced error floor due to the lack of an outer code, the $(2, 7)$-regular GLDPC code shows an important gain w.r.t. the rest of the coding schemes at small to moderate SNR values (almost 2 dB at BLER 10^{-1}). This indicates that performance can dramatically improve as long as we can accommodate an outer code that cleans up a small fraction of remaining errors. This is indeed shown to be the case in Fig. 3.10, where the $(2, 7)$-regular GLDPC code demonstrates excellent performance with comparable complexity. Furthermore, the $(2, 7)$-regular GLDPC code could be considered for coding rates up to $R = 0.287$, while the $(2, 6)$-regular GLDPC code can only go up to $R = 1/6$.

Recall that in all cases (including those in Fig. 3.10) our proposed schemes have slightly larger rate and smaller block length, yet large performance gains

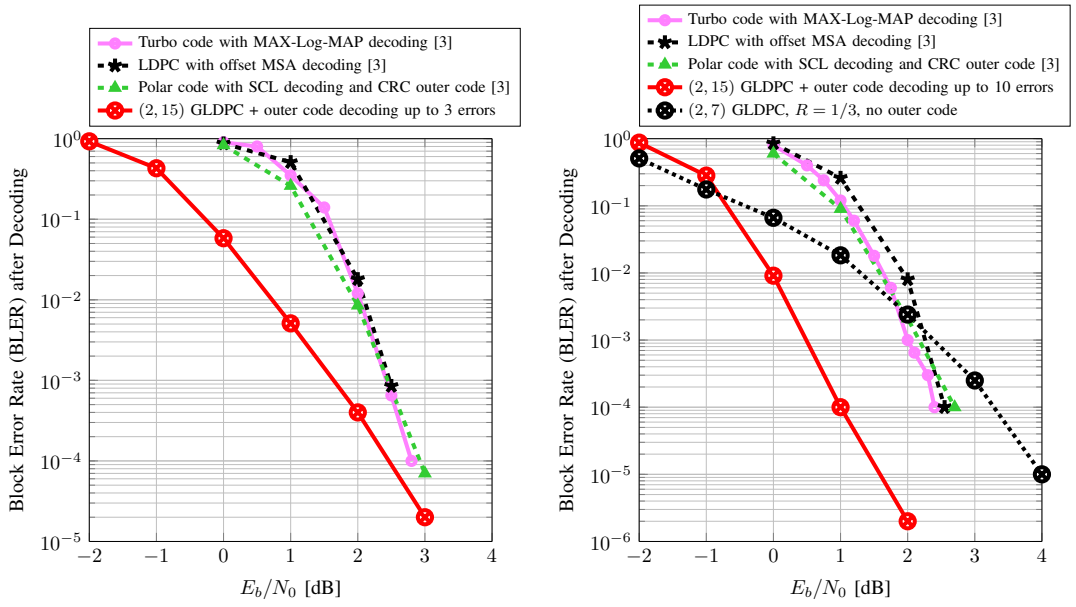


Figure 3.11: BLER over an AWGN channel with QPSK modulation for (a) the proposed rate $R = 233/465$ $(2, 15)$-regular GLDPC coding scheme (third row of Table 3.1) compared to rate $R = 1/2$ coding schemes with $M = 256$ information bits and $N = 512$ block length proposed in [28] and (b) the rate $R = 155/465$ $(2, 15)$-regular GLDPC coding scheme (last row of Table 3.1) compared to rate $R = 1/3$ coding schemes with $M = 170$ information bits and block length $N = 512$ proposed in [28]. We also include the performance of a rate $R = 0.335$ $(2, 7)$-regular GLDPC coding with $\nu = 0.667$ of GC nodes in the graph with $M = 156$ information bits, block length $N = 469$, and no outer code.

are reported. These results demonstrate the potential of the proposed design methodology: an inner GLDPC code optimized asymptotically for threshold and proportion of GC nodes, with finite length QC design based on eliminating problematic objects along with a relatively simple, off-the-shelf, hard-decision decoded outer code that cleans up the remaining errors. With such a performance gain, we believe our proposed design approach is a strong candidate for future URLLC standards.

4

Conclusions and Future work

4.1 Conclusions

In this thesis, we propose the Probabilistic Peeling Decoder algorithm as a flexible and efficient decoding algorithm that allows us to easily incorporate ML-decoded GC nodes with specific properties into the asymptotic analysis and still maintain a random definition of the graph degree distribution. Using P-PD, an asymptotic analysis of the GLDPC ensemble is carried out by a simple generalization of the original PD analysis by Luby et al. in [50]. The only information required about the component code and its decoding method is the fraction of decodable erasure patterns of a certain weight. We consider a class of GLDPC code ensembles characterized by a regular base DD where we include a certain fraction of GC nodes, and we study the tradeoff between iterative decoding threshold, coding rate and minimum distance. We have shown that one can find a fraction of GC

nodes that reduces the original gap to capacity and yields a GLDPC ensemble with linear growth of the minimum distance w.r.t. the block length. Finally, we show how the P-PD analysis can be combined with additional techniques to find a better balance between coding rate and asymptotic gap to capacity. In particular, we consider random puncturing and the use of generalized variable nodes. We would like to emphasize that, in the proposed analysis framework, the evaluation of both coding rate and of iterative decoding threshold are decoupled problems. Consequently, broader classes of component codes or improved decoding methods at GC nodes can be incorporated in a systematic way.

We further presente a novel coding scheme suited for applications such as 5G URLLC. The approach is based on combining an inner GLDPC code with a simple outer hard-decision decoded outer code (e.g., a BCH code). For this GLDPC code, the proportion of the GC nodes is optimized to guarantee an optimal asymptotic distribution (in the limit as the block length tends to infinity), while it is constructed with a simple regular quasi-cyclic graph to ensure good finite-length performance and facilitate analysis and VLSI implementation. Our results demonstrate that we can achieve remarkable gains compared to existing schemes in the literature. A $(2, 6)$-regular QC-GLDPC code, a $(2, 7)$-regular QC-GLDPC code and a $(2, 15)$-regular GLDPC code were used as examples. With the first example, we demonstrate that significant performance gains with comparable complexities (w.r.t. the state-of-the-art) can be achieved at very low coding rates. For a $(2, 15)$-regular GLDPC code, we demonstrate that these gains can also be achieved at higher rates. The naive brute-force enumeration decoding of the $(15, 11)$ component code results in a larger overall complexity than state-of-the-art competitors. However, we are confident that GLDPC ensembles with more favorable performance/complexity tradeoffs can be found using the design methodologies presented in this work, including the use of lower complexity (sparser) ensembles combined with puncturing techniques.

4.2 Future lines of work

Many practical relevant aspects have not been considered in this thesis. In the following, we provide a list of what we believe are promising lines of future work:

- Doubly-Generalized LDPC codes. The extension of the P-PD analysis to DG-GLDPC codes will allow us to find more favourable performance vs. rate trade-offs at larger coding rates.

- Efficient encoding of QC LDPC codes. It is known that QC-LDPC codes can be encoded with linear complexity (w.r.t. the blocklength) [81]. It is unknown whether the same result holds for QC GLDPC codes. Deriving efficient encoding algorithms that fully exploit the QC nature of the graph is critical for practical deployment of this kind of codes.

- Finite-length GLDPC analysis using the P-PD. In [4], the PD analysis was generalized to the finite-length regime, and scaling laws able to accurately predict the LDPC performance in the waterfall regime were derived. A priori, the same steps presented in [4] can be extended to the P-PD case. The resolution of the so-called covariance evolution equations will probably constitute the most challenging task to this end.

- Analysis of Spatially-Coupled GLDPC codes. While LDPC codes constructed from spatial coupling achieve capacity, the finite-length performance is poor, unless a moderately large number of bits are used in every position of the coupled chained. It has been demonstrated by simulation that SC-GLDPC codes can provide a more favourable tradeoff between performance and block length. The P-PD analysis framework can certainly help the code designer to find the most adequate balance between required fraction of GC nodes and desired performance.

A

Wormald's Theorem and the proof of Theorem 4

A.1 Wormald's Theorem and the proof of Theorem 4

Proving Theorem 4 is tantamount to showing that the conditions of Wormald's theorem are satisfied [103]. In this case, Theorem 4 follows directly from (A.3) and (A.4) below.

A.1.1 Wormald's theorem [103]

Let $\{Z^{(\ell)}(a)\}_{a\geq 1}$ be a d-dimensional discrete-time Markov random process with state space $\{0, 1, \ldots, \lfloor a\alpha \rfloor\}^d$ for $\alpha > 0$ and $\ell \in \mathbb{N}_+$ denotes the time index. Further let $Z_i^{(\ell)}(a)$, $i = 1, \ldots, d$ denote the i-th component of $Z^{(\ell)}(a)$. Let $\mathcal{D}$ be some open

connected bounded set containing the closure of

$$\left\{(z_1,...,z_d):\ P\left(\frac{Z_i^{(0)}(a)}{a}=z_i, 1\le i\le d\right)>0 \text{ for some } a\right\}. \tag{A.1}$$

We define the stopping time ℓ_D to be the smallest time index ℓ such that

$$(Z_1^{(\ell_D)}(a)/a,...,Z_d^{(\ell_D)}(a)/a)\notin\mathcal{D} \tag{A.2}$$

Furthermore, let $f_i(\cdot)$, $i=1,\dots,d$, be functions from $\mathbb{R}^{d+1}$ to $\mathbb{R}$. Assume that the following conditions are satisfied:

1. (Boundedness) There exists a constant ν such that for all $i=1,\dots d$, $\ell=0,\dots,\ell_{\mathcal{D}}-1$ and $a\ge 1$,
$$\left|Z_i^{(\ell+1)}(a)-Z_i^{(\ell)}(a)\right|\le\nu.$$

2. (Trend functions) For all $i=1,\dots,d$, $\ell=0,\dots,\ell_{\mathcal{D}}-1$ and $a\ge 1$,
$$\mathbb{E}\left[Z_i^{(\ell+1)}(a)/a-Z_i^{(\ell)}(a)/a\middle|Z^{(\ell)}(a)/a\right]=f_i\left(\ell/a,Z_1^{(\ell)}(a)/a,...,Z_d^{(\ell)}(a)/a\right)+\mathcal{O}(1/a).$$

3. (Lipschitz continuity) Each function $f_i(\cdot)$, $i=1,\dots,d$, is Lipschitz continuous on $\mathcal{D}$. Namely, for any pair $b,c\in\mathcal{D}$ that belongs to such intersection, there exists a constant κ such that
$$|f_i(b)-f_i(c)|\le\kappa\sum_{j=1}^{d+1}|b_j-c_j|.$$

Under these conditions, the following holds:

- The system of differential equations
$$\frac{\partial z_i}{\partial\tau}=f_i(\tau,z_1,...,z_d),\quad i=1,...,d, \tag{A.3}$$
has a unique solution for any initial condition $(b_1,...,b_d)\in\mathcal{D}$.

- There exists a strictly positive constant ζ such that
$$P\left(\left|Z_i^{(\ell)}(a)/a-z_i(\ell/a)\right|>\zeta a^{-\frac{1}{6}}\right)=\mathcal{O}\left(\mathrm{e}^{-\sqrt{a}}\right) \tag{A.4}$$
for $i=1,...,d$ and $0\le t\le t_{\mathcal{D}}$, where $z_i(\ell/a)$ is the solution to (A.3) for
$$b_i=\mathbb{E}[Z_i^{(0)}(a)]/a,\quad i=1,\dots,d. \tag{A.5}$$

The result in (A.4) states that any realization of the process $Z_i^{(t)}(a)$ concentrates around the solution predicted by (A.3) in the limit as $a \to \infty$. In the next subsection we show that this theorem is suitable to describe the expected GLDPC graph evolution of the P-PD.

A.1.2 Expected graph evolution under P-PD

To analyze the asymptotic behavior of the $\mathcal{C}_{J,K,\nu}$ ensemble under P-PD using Wormald's theorem, we identify the Markov random process $Z^{(\ell)}(a)$ in the previous section by the random process $\mathcal{G}^{(\ell)}(\mathtt{E})$, where

$$\mathcal{G}^{(\ell)}(\mathtt{E}) = \left\{ L_i^{(\ell)}, R_{pj}^{(\ell)}, R_{cj}^{(\ell)}, \hat{R}_{c\mathtt{d}}^{(\ell)}, \bar{R}_{c\mathtt{d}}^{(\ell)}, \hat{R}_{c(\mathtt{d}+1)}^{(\ell)}, \bar{R}_{c(\mathtt{d}+1)}^{(\ell)} \right\}_{\substack{i=1,\ldots,J \\ j=1,\ldots,\mathtt{d}-1,\mathtt{d}+2,\ldots,K}} \tag{A.6}$$

namely $\mathcal{G}^{(\ell)}(\mathtt{E})$ is the random process that contains all terms in the DD of the residual graph after $\ell - 1$ iterations. Note that any component in $\mathcal{G}^{(\ell)}(\mathtt{E})$ belongs to the set $\{0, 1, \ldots, \mathtt{E}\}$, and recall that $\mathtt{E}$ is the number of edges in the original GLPDC graph. Thus, $\mathtt{E}$ will play the role of the parameter a. In this subsection we prove that the evolution of $\mathcal{G}^{(\ell)}(\mathtt{E})$ under P-PD satisfies the three conditions of Wormald's theorem stated in the previous subsection. We start by computing the conditional expected evolution of all elements in $\mathcal{G}^{(\ell)}(\mathtt{E})$ after one P-PD iteration. We define the following normalized quantities:

$$\tau \triangleq \frac{\ell}{\mathtt{E}}, \quad l_i^{(\ell)} \triangleq \frac{L_i^{(\ell)}}{\mathtt{E}}, \quad r_{pj}^{(\ell)} \triangleq \frac{R_{pj}^{(\ell)}}{\mathtt{E}}, \quad r_{cj}^{(\ell)} \triangleq \frac{R_{cj}^{(\ell)}}{\mathtt{E}}, \quad \hat{r}_{c\nu}^{(\ell)} \triangleq \frac{\hat{R}_{c\nu}^{(\ell)}}{\mathtt{E}}, \quad \bar{r}_{c\nu}^{(\ell)} \triangleq \frac{\bar{R}_{c\nu}^{(\ell)}}{\mathtt{E}}, \tag{A.7}$$

for $i \in \{1, \ldots, J\}, j \in \{1, \ldots, \mathtt{d}-1, \mathtt{d}+2, \ldots, K\}$ and $\nu \in \{\mathtt{d}, \mathtt{d}+1\}$. We have that

$$r_{c\nu}^{(\ell)} = \hat{r}_{c\nu}^{(\ell)} + \bar{r}_{c\nu}^{(\ell)}, \quad \nu = \mathtt{d}, \mathtt{d}+1, \tag{A.8}$$

$$e^{(\ell)} \triangleq \sum_{i=1}^{J} l_i^{(\ell)} = \sum_{j=1}^{K} [r_{pj}^{(\ell)} + r_{cj}^{(\ell)}], \tag{A.9}$$

and $e^{(\tau)}$ is the fraction of edges remaining in the residual graph at time ℓ. The P-PD process starts at $\ell = 0$, after BEC transmission and initialization. The following relation holds between the quantities defined above at $\ell = 0$ and the $\mathcal{C}_{J,K,\nu}$ DD described in Section 2.1:

$$\mathbb{E}[l_i^{(0)}] = \epsilon \lambda_i, \tag{A.10}$$

$$\mathbb{E}[r_{pj}^{(0)}] = \sum_{\alpha \geq j} \rho_{p\alpha} \binom{\alpha - 1}{j - 1} \epsilon^j (1-\epsilon)^{\alpha - j}, \tag{A.11}$$

$$\mathbb{E}[r_{cj}^{(0)}] = \sum_{\alpha \geq j} \rho_{c\alpha} \binom{\alpha - 1}{j - 1} \epsilon^j (1-\epsilon)^{\alpha - j}, \tag{A.12}$$

for $i = 1, \dots J$ and $j = 1, \dots, K$, where the expectation is computed w.r.t. the $\mathcal{C}_{J,K,\nu}$ ensemble and the channel output. Upon initialization, every degree-$\mathtt{d}$ GC node is tagged as decodable with probability $p_{\mathtt{d}}$, and every degree-$(\mathtt{d}+1)$ GC node is tagged as decodable with probability $p_{\mathtt{d}+1}$. Recall that all GC nodes with degree less than $\mathtt{d}$ are decodable and, by assumption, all GC nodes with degree more than $\mathtt{d}+1$ are not decodable. We thus have the following initial conditions

$$\begin{aligned} \mathbb{E}[\hat{r}_{cj}^{(0)}] &= p_j \mathbb{E}[r_{cj}^{(0)}], \\ \mathbb{E}[\bar{r}_{cj}^{(0)}] &= (1-p_j) \mathbb{E}[r_{cj}^{(0)}], \quad j = \mathtt{d}, \mathtt{d}+1. \end{aligned} \tag{A.13}$$

The equations (A.10)-(A.13) correspond to the initial conditions in (A.5). Observe that since the largest GC degree is K and the largest variable node degree is J, the graph loses at most JK edges per iteration. This is an upper bound on the absolute variation of any component in $\mathcal{G}^{(\ell)}(\mathtt{E})$ between two consecutive iterations. Hence, Condition 1) of Wormald's theorem is satisfied.

Suppose we observe $\mathcal{G}^{(\ell)}(\mathtt{E})$. To derive the conditional expectations in Condition 2) of Wormald's Theorem, the so-called trend functions, we have to average among every possible scenario that we can observe after a P-PD iteration. According to Step 1) in Algorithm 3, we chose at random a decodable check node. Let $P_{p1}^{(\ell)}$ be the probability of selecting a degree-one SPC node, and let $P_{cj}^{(\ell)}$ denote the probability of selecting a decodable degree-j GC node, $j = 1, \dots, \mathtt{d}+1$. By a simple counting argument, if the check node is selected uniformly at random then

$$P_{p1}^{(\ell)} = \frac{r_{p1}^{(\ell)}}{s^{(\tau)}}, \tag{A.14}$$

$$P_{cj}^{(\ell)} = \frac{r_{cj}^{(\ell)}/j}{s^{(\tau)}}, \quad j < \mathtt{d}, \tag{A.15}$$

$$P_{cj}^{(\ell)} = \frac{\hat{r}_{cj}^{(\ell)}/j}{s^{(\tau)}}, \quad j \in \{\mathtt{d}, \mathtt{d}+1\}. \tag{A.16}$$

In (A.14)-(A.16),

$$s^{(\tau)} = r_{p1}^{(\ell)} + \sum_{w=1}^{\mathtt{d}-1} \frac{r_{cw}^{(\ell)}}{w} + \frac{\hat{r}_{c\mathtt{d}}^{(\ell)}}{\mathtt{d}} + \frac{\hat{r}_{c(\mathtt{d}+1)}^{(\ell)}}{\mathtt{d}+1} \tag{A.17}$$

is the normalized sum of decodable check nodes at the ℓ-th iteration.

Evolution of left edge degrees in the Tanner graph after one P-PD iteration

Suppose we observe the residual graph $\mathcal{G}^{(\ell)}$ at iteration ℓ. Our aim is to evaluate

$$\mathbb{E}\left[L_i^{(\ell+1)} - L_i^{(\ell)} \middle| \mathcal{G}^{(\ell)}(\mathtt{E})\right], \tag{A.18}$$

for $i = 1, 2, ..., J$. Given the graph DD $\mathcal{G}^{(\ell)}$, recall that $P_{p1}^{(\ell)}$ denotes the probability of P-PD selecting a degree-one SPC node in the current iteration, and $P_{cj}^{(\ell)}$ denotes the probability of selecting a degree-j decodable GC node. We can decompose the expectation in (A.18) according to each possible type of check node to be removed, namely,

$$\begin{aligned}\mathbb{E}\left[L_i^{(\ell+1)} - L_i^{(\ell)} \middle| \mathcal{G}^{(\ell)}(\mathtt{E})\right] &= P_{p1}^{(\ell)} \mathbb{E}\left[L_i^{(\ell+1)} - L_i^{(\ell)} \middle| \mathcal{G}^{(\ell)}(\mathtt{E}), \mathrm{Deg}_{p1}\right] \\ &\quad + \sum_{w=1}^{\mathtt{d}+1} P_{cw}^{(\ell)} \mathbb{E}\left[L_i^{(\ell+1)} - L_i^{(\ell)} \middle| \mathcal{G}^{(\ell)}(\mathtt{E}), \mathrm{Deg}_{cw}\right], \end{aligned} \tag{A.19}$$

where Deg_{p1} indicates that the P-PD removes a degree-one SPC node from the graph, and Deg_{cw} indicates that P-PD removes a degree-w decodable GC node from the graph. Computing the expectation in the first case is similar to the derivation carried out in [50] for PD with LDPC ensembles. Indeed probability that the edge adjacent to the removed degree-one SPC node has left degree i is $l_i^{(\tau)}/e^{(\tau)}$. In such a case, after deleting this variable node, the graph loses $i-1$ additional edges adjacent to this variable node, so

$$\mathbb{E}\left[L_i^{(\ell+1)} - L_i^{(\ell)} \middle| \mathcal{G}^{(\ell)}(\mathtt{E}), \mathrm{Deg}_{p1}\right] = -\frac{i l_i^{(\ell)}}{e^{(\ell)}}. \tag{A.20}$$

When the P-PD decoder removes a decodable degree-w GC node, this node is connected to w variable nodes that are also removed from the residual Tanner

graph, along with their connected edges (assuming the graph does not have double edges). Note that left degrees of the w edges connected to the removed GC node are, in general, not independent. Let $X_u \in \{1, \dots, J\}$ the RV that indicates the left degree of the u-th edge, $u = 1, \dots, w$. Arbitrarily, we can decompose the joint probability of $X_1, \dots, X_w$ as follows

$$P(X_1, \dots, X_w) = P(X_1)P(X_2|X_1)P(X_3|X_1, X_2) \dots P(X_w|X_1, \dots, X_{w-1}). \tag{A.21}$$

While $P(X_1 = x_1) = l_{x_1}^{(\ell)}/e^{(\ell)}$, $x_1 = 1, \dots, J$, the conditional distribution of X_2 given X_1 is given by

$$P(X_2 = x_2|X_1 = x_1) = \begin{cases} \dfrac{l_{x_2}^{(\tau)}}{e^{(\ell)} - 1/\mathtt{E}} & x_2 \neq x_1 \\ \\ \dfrac{l_{x_2}^{(\ell)} - 1/\mathtt{E}}{e^{(\ell)} - 1/\mathtt{E}} & x_2 = x_1 \end{cases}, \tag{A.22}$$

for $x_1, x_2 \in \{1, \dots, J\}$, where the $1/\mathtt{E}$ terms appear due to the fact that the DD has to be reparameterized after we condition on $X_1 = x_1$. The above expression can be generalized to any of the factors in (A.21) as follows:

$$\begin{aligned} P(X_u = x_u|X_1 = x_1, \dots, X_{u-1} = x_{u-1}) &= \frac{l_{x_u}^{(\ell)} - \dfrac{\sum_{u'=1}^{u} \mathbb{I}[x_{u'} = x_u]}{\mathtt{E}}}{e^{(\tau)} - \dfrac{u-1}{\mathtt{E}}} \\ &= \left(\frac{l_{x_u}^{(\ell)}}{e^{(\ell)}} - \frac{\sum_{u'=1}^{u} \mathbb{I}[x_{u'} = x_u]}{e^{(\ell)}\mathtt{E}} \right) \frac{e^{(\ell)}\mathtt{E}}{e^{(\ell)}\mathtt{E} - (u-1)}. \end{aligned} \tag{A.23}$$

Note that $e^{(\tau)}\mathtt{E}$ is the number of edges in the graph at time ℓ. Since $u \leq w < J$ and J is a constant independent of $\mathtt{E}$, the second factor in (A.23) is of order $1 - \mathcal{O}(1/\mathtt{E})$. Thus

$$P(X_1 = x_1, \dots, X_w = x_w) = \prod_{u=1}^{w} \left(\frac{l_{x_u}^{(\ell)}}{e^{(\ell)}} - \frac{\sum_{u'=1}^{u} \mathbb{I}[x_{u'} = x_u]}{e^{(\ell)}\mathtt{E}} \right) + \mathcal{O}(1/\mathtt{E}), \tag{A.24}$$

using again that $w \leq \mathtt{d} + 1 \leq J$ where J is a constant independent of $\mathtt{E}$, and that $l_{x_u}^{(\ell)}/e^{(\ell)}$ is independent of $\mathtt{E}$, we can write (A.21) as follows

$$P(X_1 = x_1, \dots, X_w = x_w) = \prod_{u=1}^{w} \frac{l_{x_u}^{(\ell)}}{e^{(\ell)}} + \mathcal{O}(1/\mathtt{E}). \tag{A.25}$$

Thus, the joint probability distribution of the left degrees of w edges connected to a degree-w GC node asymptotically factorizes as $\mathtt{E} \to \infty$ and the number of edges with left degree-i connected to the removed GC node can be roughly described by a binomial RV with parameter $l_i^{(\ell)}/e^{(\ell)}$. Hence, we obtain

$$\mathbb{E}\left[L_i^{(\ell+1)} - L_i^{(\ell)} \middle| \mathcal{G}^{(\ell)}(\mathtt{E}), \mathrm{Deg}_{cw}\right] = -\frac{iwl_i^{(\ell)}}{e^{(\tau)}} + \mathcal{O}(1/\mathtt{E}). \tag{A.26}$$

Combining (A.26) and (A.20) with (A.19), we obtain

$$\begin{aligned}\mathbb{E}\left[L_i^{(\ell+1)} - L_i^{(\ell)} \middle| \mathcal{G}^{(\ell)}(\mathtt{E})\right] &= -\frac{il_i^{(\ell)}}{e^{(\tau)}}\left(P_{p1}^{(\ell)} + \sum_{w=1}^{\mathtt{d}+1} wP_{cw}^{(\ell)}\right) + \mathcal{O}(1/\mathtt{E}) \\ &\triangleq f_i(\mathcal{G}^{(\ell)}(\mathtt{E})/\mathtt{E}) + \mathcal{O}(1/\mathtt{E}).\end{aligned} \tag{A.27}$$

Note that $f_i(\mathcal{G}^{(\ell)}(\mathtt{E})/\mathtt{E})$ depends on every component in $\mathcal{G}^{(\ell)}$, normalized by $\mathtt{E}$. Observe that $f_i(\mathcal{G}^{(\ell)}(\mathtt{E})/\mathtt{E})$ in (A.27) is of the form required by Condition 2) of Wormald's theorem.

Evolution of right edge degrees in the Tanner graph after one P-PD iteration

Our goal now is to evaluate

$$\begin{aligned}&\mathbb{E}\left[R_{pj}^{(\ell+1)} - R_{pj}^{(\ell)} \middle| \mathcal{G}^{(\ell)}(\mathtt{E})\right], \quad j = 1, \ldots, K, \\ &\mathbb{E}\left[R_{cj}^{(\ell+1)} - R_{cj}^{(\ell)} \middle| \mathcal{G}^{(\ell)}(\mathtt{E})\right], \quad j = 1, \ldots, K \text{ and } j \notin \{\mathtt{d}, \mathtt{d}+1\} \\ &\mathbb{E}\left[\hat{R}_{cj}^{(\ell+1)} - \hat{R}_{cj}^{(\ell)} \middle| \mathcal{G}^{(\ell)}(\mathtt{E})\right], \quad j \in \{\mathtt{d}, \mathtt{d}+1\} \\ &\mathbb{E}\left[\bar{R}_{cj}^{(\ell+1)} - \bar{R}_{cj}^{(\ell)} \middle| \mathcal{G}^{(\ell)}(\mathtt{E})\right], \quad j \in \{\mathtt{d}, \mathtt{d}+1\}\end{aligned}$$

As before, we evaluated these terms by conditioning on the type of check node to be removed at the current P-PD iteration. Using (A.25), the average number of edges removed from the graph after a degree-w GC node is removed is given by $\Delta_w^{(\tau)} \triangleq wa^{(\ell)} + \mathcal{O}(1/\mathtt{E})$, where $a^{(\ell)} = \sum il_i^{(\ell)}/e^{(\ell)}$. Among those, w are connected to the same degree-w GC node, i.e. they have right degree w. Consider the remaining $\Delta_w - w$ edges. Following a similar argument as in (A.25), it can be shown that the joint probability distribution of their right degree asymptotically factorizes as

$\mathtt{E} \to \infty$ and that the deviation in the finite case is dominated by $\mathcal{O}(1/\mathtt{E})$ terms. By taking $w = 1$, the same arguments hold for the case where decoder removes a degree-1 SPC node. In addition to this results, in order to evaluate the expected variation in the number of edges of certain right degree we also have to take into account that, when we remove one edge from the graph, we modify the right degree of the rest of edges still connected to the same SPC/GC node. For example, if one of the edges that are removed from the graph has right SPC degree j, after deleting such edge the graph loses j edges with right SPC degree j and gains $j-1$ edges with right SPC degree $j-1$.

Following the above arguments, conditioned on $\mathcal{G}^{(\ell)}(\mathtt{E})$, the expected change in the number of edges with right SPC degree j is given by the following expression

$$\begin{aligned}
&\mathbb{E}\left[R_{pj}^{(\ell+1)} - R_{pj}^{(\ell)} \middle| \mathcal{G}^{(\ell)}(\mathtt{E})\right] \\
&= P_{p1}^{(\ell)} \left((r_{p(j+1)}^{(\ell)} - r_{pj}^{(\ell)}) \frac{j(a(\ell)-1)}{e^{(\ell)}} - \mathbb{I}[j=1] \right) \\
&+ \sum_{w=1}^{\mathtt{d}+1} P_{cw}^{(\ell)} (r_{p(j+1)}^{(\ell)} - r_{pj}^{(\ell)}) \frac{j(wa(\ell)-w)}{e^{(\ell)}} + \mathcal{O}(1/\mathtt{E}) \\
&\triangleq g_{pj}(\mathcal{G}^{(\ell)}/\mathtt{E}) + \mathcal{O}(1/\mathtt{E}).
\end{aligned} \tag{A.28}$$

It can be further shown that the expected variation in the number of edges of right GC degree j with $j \neq \mathtt{d}, \mathtt{d}+1$ satisfies

$$\begin{aligned}
&\mathbb{E}\left[R_{cj}^{(\ell+1)} - R_{cj}^{(\ell)} \middle| \mathcal{G}^{(\ell)}(\mathtt{E})\right] \\
&= P_{p1}^{(\ell)} \left((r_{c(j+1)}^{(\ell)} - r_{cj}^{(\ell)}) \frac{j(a(\ell)-1)}{e^{(\ell)}} \right) \\
&+ \sum_{w=1}^{\mathtt{d}+1} P_{cw}^{(\ell)} \left((r_{c(j+1)}^{(\ell)} - r_{cj}^{(\ell)}) \frac{j(wa(\ell)-w)}{e^{(\ell)}} - w\mathbb{I}[j=w] \right) + \mathcal{O}(1/\mathtt{E}) \\
&\triangleq g_{cj}(\mathcal{G}^{(\ell)}/\mathtt{E}) + \mathcal{O}(1/\mathtt{E}).
\end{aligned} \tag{A.29}$$

To analyze the expected change in the number of edges connected to decodable and not decodable GC nodes of degree $\mathtt{d}$ and $\mathtt{d}+1$, we have to take into account that if a non-decodable degree-$(d+2)$ GC node loses one edge, it becomes decodable with probability $p_{\mathtt{d}+1}$. Similarly, if a non-decodable degree-$(d+1)$ GC node loses one edge, it becomes decodable with probability $p_{\mathtt{d}}$. Also note that if a decodable

GC node of degree $\mathtt{d}+1$ loses one edge, it becomes a decodable GC node of degree $\mathtt{d}$ with probability 1. It follows that the expected change in the fraction of edges connected to decodable and not decodable GC nodes of degree $j = \mathtt{d}, \mathtt{d}+1$, are given by

$$\begin{aligned}
&\mathbb{E}\left[\hat{R}_{cj}^{(\ell+1)} - \hat{R}_{cj}^{(\ell)} \middle| \mathcal{G}^{(\ell)}(\mathtt{E})\right] \\
&= P_{p1}^{(\ell)} \left((p_j \bar{r}_{c(j+1)}^{(\ell)} + \hat{r}_{c(j+1)}^{(\ell)} - \hat{r}_{cj}^{(\ell)}) \frac{j(a(\ell)-1)}{e^{(\ell)}} \right) \\
&+ \sum_{w=1}^{j+1} P_{cw}^{(\ell)} \left((p_j \bar{r}_{c(j+1)}^{(\ell)} + \hat{r}_{c(j+1)}^{(\ell)} - \hat{r}_{cj}^{(\ell)}) \frac{j(wa(\ell)-w)}{e^{(\ell)}} - w\mathbb{I}[w=j] \right) + \mathcal{O}(1/\mathtt{E}) \\
&\triangleq \hat{g}_{cj}(\mathcal{G}^{(\ell)}/\mathtt{E}) + \mathcal{O}(1/\mathtt{E})
\end{aligned} \tag{A.30}$$

$$\begin{aligned}
&\mathbb{E}\left[\bar{R}_{cj}^{(\ell+1)} - \bar{R}_{cj}^{(\ell)} \middle| \mathcal{G}^{(\ell)}(\mathtt{E})\right] \\
&= P_{p1}^{(\ell)} \left(((1-p_j) \bar{r}_{c(j+1)}^{(\ell)} - \bar{r}_{cj}^{(\ell)}) \frac{j(a(\ell)-1)}{e^{(\ell)}} \right) \\
&+ \sum_{w=1}^{j+1} P_{cw}^{(\ell)} \left(((1-p_j) \bar{r}_{c(j+1)}^{(\ell)} - \bar{r}_{cj}^{(\ell)}) \frac{j(wa(\ell)-w)}{e^{(\ell)}} - w\mathbb{I}[w=j] \right) + \mathcal{O}(1/\mathtt{E}) \\
&\triangleq \bar{g}_{cj}(\mathcal{G}^{(\ell)}/\mathtt{E}) + \mathcal{O}(1/\mathtt{E})
\end{aligned} \tag{A.31}$$

Note that $\bar{R}_{c(\mathtt{d}+2)}^{(\ell)} = R_{c(\mathtt{d}+2)}^{(\ell)}$ and $\hat{R}_{c(\mathtt{d}+2)}^{(\ell)} = 0$. Further, observe that (A.27)-(A.31) are of the form required by Condition 2) of Wormald's theorem.

On the Lipschitz continuity of the trend functions in (A.27)**-**(A.31)

Condition 3) of Wormald's theorem requires that the trend functions in (A.27)-(A.31) are Lipschitz in the set of all possible DDs. First, we note that if we would restrict the P-PD to remove only decodable check nodes (either degree-1 SPC nodes or GC nodes of one particular degree), then (A.27)-(A.31) are still valid by simply setting the corresponding probabilities $P_{p1}^{(\ell)}$ and $P_{cj}^{(\ell)}$, $j = 1, \ldots, \mathtt{d}+1$ to either zero or one. In such a case, (A.27)-(A.31) are equal up to a multiplicative constant to the PD trend functions for LDPC codes in [50], hence they are Lipschitz continuous. When we drop the restriction to remove one particular type of decodable check node, then the trend functions in (A.27)-(A.31) are convex the

combinations of Lipschitz continuous functions, with the coefficients given by the functions $P_{p1}^{(\ell)}$ and $P_{cj}^{(\ell)}$, $j = 1, \ldots, \mathtt{d}+1$ in (A.14)-(A.16), which are also Lipschitz continuous (note their similarity in form with (A.20), which is Lipschitz continuous [50]). Since they are all bounded functions, we conclude that Condition 3) of Wormald's theorem is also satisfied.

B

Proof of Theorem 5

B.1 Proof of Theorem 5

The proof of Theorem 5 closely follows that of Theorem 4 given in Appendix A.1. As before, it is sufficient to show that the conditions of Wormald's theorem are satisfied. Following the definitions given in Section 2.7.2, the left DD of the residual graph of the $\mathcal{C}_{3,K,\nu,\beta}$ code ensemble during P-PD has three components: the number of edges connected to degree-2 or degree-3 RV nodes ($L_{r2}^{(\ell)}$ and $L_{r3}^{(\ell)}$ respectively), and the number of edges connected to degree-3 GV nodes ($L_{g3}^{(\ell)}$). The right DD of the residual graph has the same elements as those defined for the $\mathcal{C}_{J,K,\nu}$ ensemble in Appendix A.1.2. Thus, the DD of the residual graph is defined

by the random process

$$\mathcal{G}^{(\ell)}(\mathtt{E}) = \left\{ L_{r2}^{(\ell)}, L_{r3}^{(\ell)}, L_{3g}^{(\ell)}, R_{pj}^{(\ell)}, R_{cj}^{(\ell)}, \hat{R}_{c\mathtt{d}}^{(\ell)}, \bar{R}_{c\mathtt{d}}^{(\ell)}, \hat{R}_{c(\mathtt{d}+1)}^{(\ell)}, \bar{R}_{c(\mathtt{d}+1)}^{(\ell)} \right\}_{j=1,\ldots,\mathtt{d}-1,\mathtt{d}+2,\ldots,K}. \tag{B.1}$$

We define

$$l_{r2}^{(\ell)} \triangleq \frac{L_{r2}^{(\ell)}}{\mathtt{E}},\ l_{r3}^{(\ell)} \triangleq \frac{L_{r3}^{(\ell)}}{\mathtt{E}},\ l_{g3}^{(\ell)} \triangleq \frac{L_{g3}^{(\ell)}}{\mathtt{E}}. \tag{B.2}$$

After P-PD initialization, i.e. $\ell = 0$, it can be shown that

$$\mathbb{E}\left[l_{g3}^{(0)}\right] = \epsilon^2\beta, \tag{B.3}$$

$$\mathbb{E}\left[l_{r3}^{(0)}\right] = \epsilon(1-\beta), \tag{B.4}$$

$$\mathbb{E}\left[l_{r2}^{(0)}\right] = 4\beta\epsilon(1-\epsilon)/3. \tag{B.5}$$

To evaluate (B.5), we compute the average number of GV nodes for which one of the two DG-LDPC coded bits is received. According to the generator matrix in (2.43), GV nodes can be viewed as degree-2 variable nodes. Based on (B.3)-(B.5), the average fraction of edges remaining in the graph after P-PD initialization is

$$\epsilon' = \epsilon(1-\beta) + 4\beta\epsilon(1-\epsilon)/3 + \epsilon^2\beta = \epsilon\left(1 + \frac{\beta(1-\epsilon)}{3}\right). \tag{B.6}$$

We can further determine expected initial conditions of the right DD of the residual graph after P-PD initialization by using (2.33) and (2.35) and replacing ϵ by ϵ'.

By following a similar procedure as in Appendix A.1.2, it can be shown that conditioned, on $\mathcal{G}^{(\ell)}(\mathtt{E})$, the expected variation in $L_{r2}^{(\ell)}, L_{r3}^{(\ell)}$, and $L_{3g}^{(\ell)}$ after one P-PD iteration is given by

$$\mathbb{E}\left[L_{r3}^{(\ell+1)} - L_{r3}^{(\ell)} \middle| \mathcal{G}^{(\ell)}\right] = -\frac{3l_{r3}^{(\ell)}}{e^{(\ell)}}\left(P_{p1}^{(\ell)} + \sum_{w=1}^{\mathtt{d}+1} wP_{cw}^{(\ell)}\right) + \mathcal{O}(1/\mathtt{E}), \tag{B.7}$$

$$\mathbb{E}\left[L_{r2}^{(\ell+1)} - L_{r2}^{(\ell)} \middle| \mathcal{G}^{(\ell)}\right] = \left(\frac{2l_{g3}^{(\ell)}}{e^{(\ell)}} - \frac{2l_{r2}^{(\ell)}}{e^{(\ell)}}\right)\left(P_{p1}^{(\ell)} + \sum_{w=1}^{\mathtt{d}+1} wP_{cw}^{(\ell)}\right) + \mathcal{O}(1/\mathtt{E}), \tag{B.8}$$

$$\mathbb{E}\left[L_{g3}^{(\ell+1)} - L_{g3}^{(\ell)} \middle| \mathcal{G}^{(\ell)}\right] = -\frac{3l_{g3}^{(\ell)}}{e^{(\ell)}}\left(P_{p1}^{(\ell)} + \sum_{w=1}^{\mathtt{d}+1} wP_{cw}^{(\ell)}\right) + \mathcal{O}(1/\mathtt{E}), \tag{B.9}$$

where $e(\ell) = l_{r3}^{(\ell)} + l_{g3}^{(\ell)} + l_{g3}^{(\ell)}$ and $P_{p1}^{(\ell)}$ and $P_{cw}^{(\ell)}$ are given in (2.29) and (2.30) respectively. In (B.8), we have used that that if a degree-3 GV node loses one edge,

then the graph loses 3 edges with left GV degree 3 and gains 2 edges with left RV degree 2. The conditional expected variation of the right DD of the residual graph can be computed using (A.28)-(A.31) by taking $a^{(\ell)} = (3l_{r3}^{(\ell)} + 2l_{r2}^{(\ell)} + l_{g3}^{(\ell)})/e^{(\ell)}$. Finally, proving that the conditions in Wormald's Theorem hold follows by the same arguments as in the proof of Theorem 4 in Appendix A.1.

C

Generator matrices of reference Codes

C.1 Generator matrices of reference Codes

Reference codes have been found by performing an exhaustive search over the database [29, 30], which implements MAGMA [8] to design block codes with the largest minimum distance.

Code R-I: Rate-1/2 Hamming $(6, 3)$ linear block code with generator matrix

$$\mathrm{G}_{\text{R-I}} = \begin{pmatrix} 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 \end{pmatrix} \tag{C.1}$$

Code R-II: Rate-1/3 Cordaro-Wagner 2-dimensional repetition code of length

6 with generator matrix

$$\mathrm{G}_{\text{R-II}} = \begin{pmatrix} 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 \end{pmatrix} \tag{C.2}$$

Code R-III: Rate-4/7 Hamming (7,4) code with generator matrix

$$\mathrm{G}_{\text{R-III}} = \begin{pmatrix} 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 \end{pmatrix} \tag{C.3}$$

Code R-IV: Rate-3/7 linear block code with generator matrix

$$\mathrm{G}_{\text{R-IV}} = \begin{pmatrix} 0 & 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 & 1 \end{pmatrix} \tag{C.4}$$

Code R-V: Rate-1/2 extended $(7, 4)$-Hamming code with extra parity bit, i.e., $(8, 4)$ Hamming code. Another example is a Quasi-Cyclic $(8, 4, 4)$ code with generator matrix

$$\mathrm{G}_{\text{R-V}} = \begin{pmatrix} 1 & 0 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 & 1 & 0 \end{pmatrix} \tag{C.5}$$

Code R-VI: Rate-3/8 cyclic linear block code with generator matrix

$$\mathrm{G}_{\text{R-VI}} = \begin{pmatrix} 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 \end{pmatrix} \tag{C.6}$$

Code R-VII: Rate-1/4 Cordaro-Wagner 2-dimensional repetition code of length 8 with generator matrix

$$\mathrm{G}_{\text{R-VII}} = \begin{pmatrix} 1 & 0 & 1 & 1 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 1 & 1 \end{pmatrix} \tag{C.7}$$

Code R-VIII: Rate-11/15 linear block code with generator matrix

$$\mathrm{G}_{\text{R-VIII}} = \begin{pmatrix} 0 & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} \tag{C.8}$$

Code R-IX: Rate-2/3 linear block code with generator matrix

$$G_{\text{R-IX}} = \begin{pmatrix} 0 & 1 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \end{pmatrix} \tag{C.9}$$

D

Update Rules of the GC Nodes for the (6,3) Shortened Hamming code

D.1 Update Rules of the GC Nodes for the (6,3) Shortened Hamming code

Let vector (p_{i0}, p_{i1}) denote the probabilistic input message coming from the ith variable connected to the GC node, where $p_{i0} + p_{i1} = 0$ and index i, $i = 1, 2, \ldots, 6$, corresponds to the ith position of the component code. Similarly, we denote by $(\tilde{p}_{i0}, \tilde{p}_{i1})$ the probabilistic output message extrinsic probabilities from the GC node to the ith variable node connected to it.

According to the (6,3) Hamming codebook in (3.9), we can check by enumer-

ation that, for instance, we can compute $\tilde{p}_{10}$ and $\tilde{p}_{11}$ as follows

$$\begin{aligned}\tilde{p}_{10} &= p_{20}p_{30}p_{40}p_{50}p_{60} + p_{20}p_{31}p_{40}p_{51}p_{61} + p_{21}p_{30}p_{41}p_{50}p_{61} \\ &\quad + p_{21}p_{31}p_{41}p_{51}p_{60}, \\ \tilde{p}_{11} &= p_{20}p_{30}p_{41}p_{51}p_{60} + p_{20}p_{31}p_{41}p_{50}p_{61} + p_{21}p_{30}p_{40}p_{51}p_{61} \\ &\quad + p_{21}p_{31}p_{40}p_{50}p_{60}. \end{aligned} \tag{D.1}$$

Let $\tilde{\lambda}_{p1} = \tilde{p}_{10}/\tilde{p}_{11}$ and $\tilde{\Lambda}_{p1} = \log(\tilde{\lambda}_{p1})$, thus

$$\begin{aligned}&\tilde{\Lambda}_{p1} = \\ &\quad \log(\mathrm{e}^{\Lambda_{p2}+\Lambda_{p3}+\Lambda_{p4}+\Lambda_{p5}+\Lambda_{p6}} + \mathrm{e}^{\Lambda_{p2}+\Lambda_{p4}} + \mathrm{e}^{\Lambda_{p3}+\Lambda_{p5}} + \mathrm{e}^{\Lambda_{p6}}) \\ &- \log(\mathrm{e}^{\Lambda_{p2}+\Lambda_{p3}+\Lambda_{p6}} + \mathrm{e}^{\Lambda_{p2}+\Lambda_{p5}} + \mathrm{e}^{\Lambda_{p3}+\Lambda_{p4}} + \mathrm{e}^{\Lambda_{p4}+\Lambda_{p5}+\Lambda_{p6}})\end{aligned} \tag{D.2}$$

which is decided by the biggest exponent of the subtrahend and the minuend using the log-sum expression. The expression in (D.2) can be easily generated for all output messages in the GC nodes as follows. Let $\mathbf{C}_{i,m}^{(6,3)}$ denote the m-th bit of the i-th codeword, where $\mathbf{C}_{i,m}^{(6,3)}$ is given in (3.9). Then we have

$$\begin{aligned}\tilde{\Lambda}_j = &\log\left[\sum_{\substack{i\in\{1,8\}\\ C_{i,j}=0}} \exp\left(\sum_{\substack{m\in\{1,6\}\\ m\neq j}} \mathbb{I}[(\mathbf{C}_{i,m}^{(6,3)}=0)](\Lambda_{pj}-\Lambda^*)\right)\right] \\ &- \log\left[\sum_{\substack{i\in\{1,8\}\\ C_{i,j}=1}} \exp\left(\sum_{\substack{m\in\{1,6\}\\ m\neq j}} \mathbb{I}[(\mathbf{C}_{i,m}^{(6,3)}=1)](\Lambda_{pj}-\Lambda^*)\right)\right].\end{aligned} \tag{D.3}$$

References

[1] S. Abu-Surra, D. Divsalar, and W. E. Ryan. Enumerators for protograph-based ensembles of LDPC and generalized LDPC codes. *IEEE Trans. on Inf. Theory*, 57(2):858–886, Feb. 2011.

[2] Shadi Abu-Surra, William E Ryan, and Dariush Divsalar. Ensemble enumerators for protograph-based generalized LDPC codes. In *Global Telecommunications Conference, 2007. GLOBECOM'07. IEEE*, pages 1492–1497. IEEE, 2007.

[3] Noga Alon and Michael Luby. A linear time erasure-resilient code with nearly optimal recovery. *IEEE Transactions on Information Theory*, 42(6):1732–1736, 1996.

[4] A. Amraoui, A. Montanari, T. Richardson, and R. Urbanke. Finite-length scaling for iteratively decoded LDPC ensembles. *IEEE Trans. on Inf. Theory*, 55(2):473 –498, Feb. 2009.

[5] Kenneth S Andrews, Dariush Divsalar, Sam Dolinar, Jon Hamkins, Christopher R Jones, and Fabrizio Pollara. The development of turbo and LDPC codes for deep-space applications. *Proceedings of the IEEE*, 95(11):2142–2156, 2007.

[6] A. Ashikhmin and S. Litsyn. Simple MAP decoding of first-order Reed-Muller and Hamming codes. *IEEE Trans. on Information Theory*, 50(8):1812–1818, Aug. 2004.

[7] Alexios Balatsoukas-Stimming, Pascal Giard, and Andreas Burg. Comparison of polar decoders with existing low-density parity-check and turbo decoders. In *Wireless Communications and Networking Conference Workshops (WCNCW), 2017*, pages 1–6. IEEE, 2017.

[8] Wieb Bosma, John Cannon, and Catherine Playoust. The Magma Algebra System I: The User Language. *Journal of Symbolic Computation*, 24(3-4):235–265, October 1997.

[9] J Boutros, O Pothier, and G Zemor. Generalized low density (Tanner) codes. In *Communications, 1999. ICC'99. 1999 IEEE International Conference on*, volume 1, pages 441–445. IEEE, 1999.

[10] D. Burshtein and G. Miller. Efficient maximum-likelihood decoding of LDPC codes over the binary erasure channel. *IEEE Trans. on Inf. Theory*, 50(11):2837 – 2844, Nov. 2004.

[11] He Chen, Rana Abbas, Peng Cheng, Mahyar Shirvanimoghaddam, Wibowo Hardjawana, Wei Bao, Yonghui Li, and Branka Vucetic. Ultra-reliable low latency cellular networks: use cases, challenges and approaches. *arXiv preprint arXiv:1709.00560*, 2017.

[12] Lei Chen, Ivana Djurdjevic, and Jun Xu. Construction of quasicyclic LDPC codes based on the minimum weight codewords of Reed-Solomon codes. In *IEEE International Symposium on Information Theory, Chicago, USA*, page 239, 2004.

[13] Lei Chen, Jun Xu, I. Djurdjevic, and S. Lin. Near-Shannon-limit quasi-cyclic low-density parity-check codes. *IEEE Trans. on Communications*, 52(7):1038–1042, July 2004.

[14] Pin-Han Chen, Jian-Jia Weng, Chung-Hsuan Wang, and Po-Ning Chen. BCH code selection and iterative decoding for BCH and LDPC concatenated coding system. *IEEE Communications Letters*, 17(5):980–983, 2013.

[15] Shanzhi Chen and Jian Zhao. The requirements, challenges, and technologies for 5G of terrestrial mobile telecommunication. *IEEE Communications Magazine*, 52(5):36–43, 2014.

[16] Sae-Young Chung, G David Forney, Thomas J Richardson, and Rüdiger Urbanke. On the design of low-density parity-check codes within 0.0045 dB of the Shannon limit. *IEEE Communications letters*, 5(2):58–60, 2001.

[17] Daniel J Costello. *Error Control Coding: Fundamentals and Applications*. Prentice Hall, 1983.

[18] Zhiqiang Cui, Zhongfeng Wang, and Xinmiao Zhang. Reduced-complexity column-layered decoding and implementation for ldpc codes. *IET communications*, 5(15):2177–2186, 2011.

[19] Ivan B Djordjevic, Olgica Milenkovic, and Bane Vasic. Generalized low-density parity-check codes for optical communication systems. *Journal of Lightwave Technology*, 23(5):1939–1946, 2005.

[20] Ivan B Djordjevic and Ting Wang. Multiple component codes based generalized LDPC codes for high-speed optical transport. *Optics express*, 22(14):16694–16705, 2014.

[21] Catherine Douillard, Michel Jézéquel, Claude Berrou, Nathalie Brengarth, Jacky Tousch, and Nghia Pham. The turbo code standard for DVB-RCS. In *2nd International Symposium on Turbo Codes & Related Topics, Brest, France*, pages 535–538, 2000.

[22] E. Paolini, M.P.C. Fossorier and M. Chiani. Generalized and Doubly Generalized LDPC Codes With Random Component Codes for the Binary Erasure Channel. *IEEE Trans. on Inf. Theory*, 56(4):1651–1672, Apr. 2010.

[23] Kjetil Fagervik and Arne Sjøthun Larssen. Performance and complexity comparison of low density parity check codes and turbo codes. In *Proc. Norwegian Signal Processing Symposium,(NORSIG'03)*, pages 2–4, 2003.

[24] M. F. Flanagan, E. Paolini, M. Chiani, and M. P. C. Fossorier. Spectral Shape of Doubly-Generalized LDPC Codes: Efficient and Exact Evaluation. *IEEE Transactions on Information Theory*, 59(11):7212–7228, Nov 2013.

[25] Marc PC Fossorier. Quasi cyclic low-density parity-check codes from circulant permutation matrices. *IEEE Trans. on Inf. Theory*, 50(8):1788–1793, 2004.

[26] R. G. Gallager. *Low Density Parity Check Codes*. MIT Press, 1963.

[27] Robert Gallager. Low-density parity-check codes. *IRE Transactions on information theory*, 8(1):21–28, 1962.

[28] Heshani Gamage, Nandana Rajatheva, and Matti Latva-aho. Channel coding for enhanced mobile broadband communication in 5G systems. In *IEEE European Conf. on Networks and Communications (EuCNC), Oulu, Finland*, pages 1–6. IEEE, 2017.

[29] Markus Grassl. Bounds on the minimum distance of linear codes and quantum codes. Online available at `http://www.codetables.de`. Accessed on 2017-01-07.

[30] Markus Grassl. Searching for linear codes with large minimum distance. In Wieb Bosma and John Cannon, editors, *Discovering Mathematics with Magma — Reducing the Abstract to the Concrete*, volume 19 of *Algorithms and Computation in Mathematics*, pages 287–313. Springer, Heidelberg, 2006.

[31] R. Guan and L. Zhang. Hybrid hamming GLDPC codes over the binary erasure channel. In *2017 11th IEEE International Conference on Anti-counterfeiting, Security, and Identification (ASID)*, pages 130–133, Oct 2017.

[32] Manabu Hagiwara, Kenta Kasai, Hideki Imai, and Kohichi Sakaniwa. Spatially coupled quasi-cyclic quantum LDPC codes. In *Information Theory Proceedings (ISIT), 2011 IEEE International Symposium on*, pages 638–642. IEEE, 2011.

[33] Yejun He, Jie Yang, and Jiawei Song. A survey of error floor of LDPC codes. In *Communications and Networking in China (CHINACOM), 2011 6th International ICST Conference on*, pages 61–64. IEEE, 2011.

[34] William Cary Huffman and Vera Pless. *Fundamentals of error-correcting codes*. Cambridge University Press, Cambridge, U.K., New York, 2003.

[35] Nadine Hussami, Satish Babu Korada, and Rudiger Urbanke. Performance of polar codes for channel and source coding. In *Information Theory, 2009. ISIT 2009. IEEE International Symposium on*, pages 1488–1492. IEEE, 2009.

[36] Y. Y. Jian, H. D. Pfister, and K. R. Narayanan. Approaching capacity at high rates with iterative hard-decision decoding. In *Proc. IEEE ISIT, Boston, USA*, pages 2696–2700, July 2012.

[37] T. Kasami, T. Takata, T. Fujiwara, and S. Lin. On the optimum bit orders with respect to the state complexity of trellis diagrams for binary linear codes. *IEEE Trans. on Information Theory*, 39(1):242–245, Jan. 1993.

[38] Haesik Kim. Coding and modulation techniques for high spectral efficiency transmission in 5G and Satcom. In *Signal Processing Conference (EUSIPCO), 2015 23rd European*, pages 2746–2750. IEEE, 2015.

[39] Ioannis Kouretas, Charalambos Basetas, and Vassilis Paliouras. Low-power logarithmic number system addition/subtraction and their impact on digital filters. *IEEE Trans. on Computers*, 62(11):2196–2209, 2013.

[40] Frank R Kschischang, Brendan J Frey, and H-A Loeliger. Factor graphs and the sum-product algorithm. *IEEE Trans. on Information Theory*, 47(2):498–519, 2001.

[41] Stefan Laendner and Olgica Milenkovic. LDPC codes based on latin squares: Cycle structure, stopping set, and trapping set analysis. *IEEE Transactions on Communications*, 55(2):303–312, 2007.

[42] Martin Langhammer and Bogdan Pasca. Single Precision Logarithm and Exponential Architectures for Hard Floating-Point Enabled FPGAs. *IEEE Trans. on Computers*, 66(12):2031–2043, 2017.

[43] M. Lentmaier and G.P. Fettweis. On the thresholds of generalized LDPC convolutional codes based on protographs. In *IEEE International Symposium on Information Theory 2010, Austin, USA*, pages 709–713, June.

[44] M. Lentmaier and K.Sh. Zigangirov. On generalized low-density parity-check codes based on Hamming component codes. *IEEE Communications Letters*, 3(8):248–250, Aug. 1999.

[45] Zongwang Li, Lei Chen, Lingqi Zeng, Shu Lin, and Wai H Fong. Efficient encoding of quasi-cyclic low-density parity-check codes. *IEEE Trans. on Communications*, 54(1):71–81, 2006.

[46] Yanfang Liu, Pablo M. Olmos, and Tobias Koch. A probabilistic peeling decoder to efficiently analyze generalized LDPC codes over the BEC. *arXiv preprint arXiv:1709.00873*, abs/1709.00873, 2017.

[47] Gianluigi Liva and Marco Chiani. Protograph LDPC codes design based on EXIT analysis. In *Global Telecommunications Conference, 2007. GLOBECOM'07. IEEE*, pages 3250–3254. IEEE, 2007.

[48] Gianluigi Liva, W.E. Ryan, and M. Chiani. Quasi-cyclic generalized LDPC codes with low error floors. *IEEE Trans. on Communications*, 56(1):49–57, Jan. 2008.

[49] Gianluigi Liva, William E Ryan, and Marco Chiani. Quasi-cyclic generalized LDPC codes with low error floors. *IEEE Transactions on Communications*, 56(1), 2008.

[50] M.G. Luby, M. Mitzenmacher, M.A. Shokrollahi, and D.A. Spielman. Efficient erasure correcting codes. *IEEE Trans. on Inf. Theory*, 47(2):569 –584, Feb. 2001.

[51] M.G. Luby, M. Mitzenmacher, M.A. Shokrollahi, and D.A. Spielman. Improved low-density parity-check codes using irregular graphs. *IEEE Trans. on Inf. Theory*, 47(2):585 –598, Feb. 2001.

[52] David J. C. MacKay. *Inf. Theory, Inference, and Learning Algorithms*. Cambridge University Press, 2003.

[53] David JC MacKay. Good error-correcting codes based on very sparse matrices. *IEEE transactions on Information Theory*, 45(2):399–431, 1999.

[54] David JC MacKay and Matthew C Davey. Evaluation of gallager codes for short block length and high rate applications. In *Codes, Systems, and Graphical Models*, pages 113–130. Springer, 2001.

[55] David JC MacKay and Radford M Neal. Good codes based on very sparse matrices. In *IMA International Conference on Cryptography and Coding*, pages 100–111. Springer, 1995.

[56] Florence Jessie MacWilliams and Neil James Alexander Sloane. *The theory of error correcting codes*. North-Holland Pub. Co. New York, 1977.

[57] C. Measson, A. Montanari, and R. Urbanke. Maxwell construction: The hidden bridge between iterative and maximum a posteriori decoding. *IEEE Trans. on Inf. Theory*, 54(12):5277 –5307, Dec. 2008.

[58] Reddipogu Nissi Merlyn and EV Narayana. Parallel Folded Architecture for Encoding and Decoding the Polar (16, k) Code.

[59] N. Miladinovic and M.P.C. Fossorier. Generalized LDPC codes and generalized stopping sets. *IEEE Trans. on Communications*, 56(2):201–212, February 2008.

[60] Nenad Miladinovic and Marc Fossorier. Generalized LDPC codes with Reed-Solomon and BCH codes as component codes for binary channels. In *Global Telecommunications Conference, 2005. GLOBECOM'05. IEEE*, volume 3, pages 6–pp. IEEE, 2005.

[61] Nenad Miladinovic and Marc PC Fossorier. Generalized LDPC codes and generalized stopping sets. *IEEE Transactions on Communications*, 56(2), 2008.

[62] David GM Mitchell, Michael Lentmaier, and Daniel J Costello. Spatially coupled LDPC codes constructed from protographs. *IEEE Transactions on Information Theory*, 61(9):4866–4889, 2015.

[63] David GM Mitchell, Ali E Pusane, Kamil Sh Zigangirov, and Daniel J Costello. Asymptotically good LDPC convolutional codes based on protographs. In *Information Theory, 2008. ISIT 2008. IEEE International Symposium on*, pages 1030–1034. IEEE, 2008.

[64] David GM Mitchell, Roxana Smarandache, and Daniel J Costello. Quasi-cyclic LDPC codes based on pre-lifted protographs. *IEEE Transactions on Information Theory*, 60(10):5856–5874, 2014.

[65] D.G.M. Mitchell, M. Lentmaier, and D.J. Costello. On the minimum distance of generalized spatially coupled LDPC codes. In *Proc. IEEE Int. Symp. Inf. Theory (ISIT), Istanbul, Turkey*, pages 1874–1878, July 2013.

[66] D.G.M. Mitchell, M. Lentmaier, A.E. Pusane, and D.J. Costello. Randomly Punctured LDPC Codes. *IEEE Journal on Selected Areas in Communications*, 34(2):408–421, Feb 2016.

[67] Todd K Moon. Error correction coding. *Mathematical Methods and Algorithms. Jhon Wiley and Son*, 2005.

[68] I. P. Mulholland, E. Paolini, and M. F. Flanagan. Design of ldpc code ensembles with fast convergence properties. In *IEEE International Black Sea Conf. on Communications and Networking, Constanta, Romania*, pages 53–57, May 2015.

[69] B. Müller, M. Holters, and U. Zölzer. Low complexity Soft-Input Soft-Output Hamming Decoder. In *IEEE Federation of Telecommunications Engineers*

of the European Union (FITCE) Congress, Palermo, Italy, pages 1–5, Aug. 2011.

[70] Wooseok Nam, Dongwoon Bai, Jungwon Lee, and Inyup Kang. Advanced interference management for 5G cellular networks. *IEEE Communications Magazine*, 52(5):52–60, 2014.

[71] Md Noor-A-Rahim, Khoa D Nguyen, and Gottfried Lechner. Finite length analysis of LDPC codes. In *Wireless Communications and Networking Conference (WCNC), 2014 IEEE*, pages 206–211. IEEE, 2014.

[72] Daesun Oh and Keshab K Parhi. Optimally quantized offset min-sum algorithm for flexible LDPC decoder. In *IEEE Asilomar Conf. on Signals, Systems & Computers, Pacific Grove, USA*, pages 1886–1891, 2008.

[73] Pablo M Olmos, David GM Mitchell, Dmitri Truhachev, and Daniel J Costello. Continuous Transmission of Spatially Coupled LDPC Code Chains. *IEEE Transactions on Communications*, 65(12):5097–5109, 2017.

[74] Pablo M Olmos and Ruediger L Urbanke. A scaling law to predict the finite-length performance of spatially-coupled LDPC codes. *IEEE Transactions on Information Theory*, 61(6):3164–3184, 2015.

[75] P.M. Olmos, D.G.M. Mitchell, and Jr. Costello, D.J. Analyzing the finite-length performance of generalized LDPC codes. In *Proc. IEEE ISIT, Hong Kong, China*, pages 2683–2687, June 2015.

[76] Alon Orlitsky, Krishnamurthy Viswanathan, and Junan Zhang. Stopping set distribution of LDPC code ensembles. *IEEE Transactions on Information Theory*, 51(3):929–953, 2005.

[77] E. Paolini, M. Fossorier, and M. Chiani. On the design of irregular GLDPC codes with low error floor over the BEC. In *2008 International Symposium on Information Theory and Its Applications*, pages 1–6, Dec 2008.

[78] Enrico Paolini, Marc Fossorier, and Marco Chiani. Generalized stability condition for generalized and doubly-generalized LDPC codes. In *Information Theory, 2007. ISIT 2007. IEEE International Symposium on*, pages 1536–1540. IEEE, 2007.

[79] Y. Polyanskiy, H.V. Poor, and S. Verdu. Channel coding rate in the finite blocklength regime. *IEEE Trans. on Inf. Theory*, 56(5):2307 –2359, May 2010.

[80] Petar Popovski. Ultra-reliable communication in 5G wireless systems. In *IEEE Int. Conf. on 5G for Ubiquitous Connectivity (5GU), Nanjing, China.*, pages 146–151. IEEE, 2014.

[81] Hanghang Qi and Norbert Goertz. Low-Complexity Encoding of LDPC Codes: A New Algorithm and its Performance. *Institute for Digital Communications, Joint Research Institute for Signal & Image Processing, School of Engineering and Electronics, The University of Edinburgh*, 2007.

[82] T.J. Richardson, Amin Shokrollahi, and R. Urbanke. Design of capacity-approaching irregular low-density parity-check codes. *IEEE Trans. on Inf. Theory*, 47(2):619 –637, Feb. 2001.

[83] T.J. Richardson and R. Urbanke. The capacity of low-density parity-check codes under message-passing decoding. *IEEE Trans. on Inf. Theory*, 47(2):599 –618, Feb. 2001.

[84] Tom Richardson. Error floors of LDPC codes. In *Proceedings of the annual Allerton conference on communication control and computing*, volume 41, pages 1426–1435. The University; 1998, 2003.

[85] Tom Richardson and Ruediger Urbanke. *Modern coding theory*. Cambridge university press, 2008.

[86] Tom J. Richardson and Ruediger Urbanke. *Modern Coding Theory*. Cambridge University Press, Mar. 2008.

[87] Gerd Richter. Finding small stopping sets in the Tanner graphs of LDPC codes. In *Proc. Int. Symp. on Turbo Codes Related Topics and Int. ITG-Conf. on Source and Channel Coding, Munich, Germany*, pages 1–5. Citeseer, 2006.

[88] William Ryan and Shu Lin. *Channel codes: classical and modern*. Cambridge University Press, 2009.

[89] Shin-Lin Shieh. Concatenated BCH and LDPC coding scheme with iterative decoding algorithm for flash memory. *IEEE communications letters*, 19(3):327–330, 2015.

[90] Mahyar Shirvanimoghaddam, Mohamad Sadegh Mohamadi, Rana Abbas, Aleksandar Minja, Balazs Matuz, Guojun Han, Zihuai Lin, Yonghui Li, Sarah Johnson, and Branka Vucetic. Short Block-length Codes for Ultra-Reliable Low-Latency Communications. *arXiv preprint arXiv:1802.09166*, 2018.

[91] Keattisak Sripimanwat. *Turbo code applications*, volume 1. Springer, 2005.

[92] Michal Sybis, Krzysztof Wesolowski, Keeth Jayasinghe, Venkatkumar Venkatasubramanian, and Vladimir Vukadinovic. Channel coding for ultra-reliable low-latency communication in 5G systems. In *Vehicular Technology Conference (VTC-Fall), 2016 IEEE 84th*, pages 1–5. IEEE, 2016.

[93] Ying Yu Tai, Lan Lan, Lingqi Zeng, Shu Lin, and Khaled AS Abdel-Ghaffar. Algebraic construction of quasi-cyclic LDPC codes for the AWGN and erasure channels. *IEEE Trans. on Communications*, 54(10):1765–1774, 2006.

[94] Heng Tang, Jun Xu, Shu Lin, and Khaled AS Abdel-Ghaffar. Codes on finite geometries. *IEEE Transactions on Information Theory*, 51(2):572–596, 2005.

[95] R.M. Tanner. A recursive approach to low complexity codes. *IEEE Trans. on Inf. Theory*, 27(5):533 – 547, Sept. 1981.

[96] Jeremy Thorpe. Low-density parity-check (LDPC) codes constructed from protographs. *IPN progress report*, 42(154):42–154, 2003.

[97] Jeremy Thorpe. Low density parity check (LDPC) codes constructed from protographs. Technical report, JPL IPN Progress Report 42-154, 2003.

[98] B Vasic and EM Kurtas. An introduction to LDPC codes. In *CRC Handbook for Coding and Signal Processing for Recording Systems*. CRC, 2004.

[99] Pascal O Vontobel and Ralf Koetter. Lower bounds on the minimum pseudoweight of linear codes. In *Information Theory, 2004. ISIT 2004. Proceedings. International Symposium on*, page 70. IEEE, 2004.

[100] Pascal O Vontobel and Ralf Koetter. Graph-cover decoding and finite-length analysis of message-passing iterative decoding of LDPC codes. *arXiv preprint cs/0512078*, 2005.

[101] Y. Wang and M. Fossorier. Doubly Generalized LDPC Codes. In *2006 IEEE International Symposium on Information Theory*, pages 669–673, July 2006.

[102] Yige Wang and M. Fossorier. Doubly Generalized LDPC Codes over the AWGN Channel. *IEEE Trans. on Communications*, 57(5):1312–1319, May 2009.

[103] Nicholas C. Wormald. Differential equations for random processes and random graphs. *Annals of Applied Probability*, 5(4):1217–1235, 1995.

[104] Xin Xiao, Mona Nasseri, Bane Vasić, and Shu Lin. Serial concatenation of reed muller and LDPC codes with low error floor. In *55th Annual Allerton Conference on Communication, Control, and Computing, Allerton, USA*, pages 688–693. IEEE, 2017.

[105] Ningde Xie, Wei Xu, Tong Zhang, Erich F Haratsch, and Jaekyun Moon. Concatenated low-density parity-check and BCH coding system for magnetic recording read channel with 4 kb sector format. *IEEE Transactions on Magnetics*, 44(12):4784–4789, 2008.

[106] Jun Xu, Lei Chen, Lingqi Zeng, Lan Lan, and Shu Lin. Construction of low-density parity-check codes by superposition. *IEEE Trans. on Communications*, 53(2):243–251, 2005.

[107] Guosen Yue, Li Ping, and Xiaodong Wang. Generalized Low-Density Parity-Check Codes Based on Hadamard Constraints. *IEEE Trans. on Inf. Theory,*, 53(3):1058–1079, Mar. 2007.

[108] Guosen Yue, Li Ping, and Xiaodong Wang. Generalized low-density parity-check codes based on Hadamard constraints. *IEEE Transactions on Information Theory*, 53(3):1058–1079, 2007.

[109] Junan Zhang and Alon Orlitsky. Finite-length analysis of LDPC codes with large left degrees. In *Information Theory, 2002. Proceedings. 2002 IEEE International Symposium on*, page 3. IEEE, 2002.

[110] Jianguang Zhao, F. Zarkeshvari, and A. H. Banihashemi. On implementation of min-sum algorithm and its modifications for decoding low-density parity-check (ldpc) codes. *IEEE Trans. on Communications*, 53(4):549–554, Apr. 2005.